5. Public Class Descriptions

The following class descriptions are grouped by the CSCI that exports that class. A subsection is devoted to each CSCI or class category. Within each, public classes (those classes that export services outside of their own CSCI) are described along with their public operations and attributes. The details on the class designs and non-public class descriptions are found in the individual subsystem design specifications (DID 305 subdocuments).

The overview subdocument of the design specification [305-CD-020-002] provides the context for the CSCIs shown here.

5.1 Advertising Service Classes

5.1.1 Advertising Service Classes Overview

The Advertising Service provides the interfaces needed to support Client defined interactive browsing and searching of advertisements. Although there will be a single format for submitting advertisements to the service, advertisements should be accessible via several different interfaces to support database searching, text searching, and hyperlinked access and retrieval according to several different viewing styles (e.g., plain ASCII text, interactive form, or HTML document). A data server or other provider will advertise its data collections and services with the Advertising Service. The advertisement will include a listing of all products (and other Earth Science Data Types) available in the collection and a set of product attributes. Advertisements include directory level metadata, therefore,, the attributes reflected in the advertising service include the ECS Core Metadata Directory-Level attributes. The workbench will send user queries which access only directory level metadata directly to the advertising service (rather than sending it as a distributed query to the various sites which provided the advertising information). A user who wishes to find out what data sets are available on the network can search (i.e., formulate a query) or browse (i.e., navigate through hyperlinked pages of advertisements) the advertising information. Both types of 'directory searching' are available on the user's desktop; the user can choose whichever approach is most convenient in the current work context. Since the ADSRV CSCI is on the incremental track of development, requirements, schedule, scenarios, issues and design are documented in a Software Development File (SDF) for ADSRV.

In Release B, there should be no major changes to the Release A public classes.

5.1.2 Advertising Service Class Descriptions

5.1.2.1 Class EcPoHandle

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcPoPersistentBase(Public) Coordinate

Description:

Public View: This class works in concert with EcPoPersistentBase to form the abstract base classes for a persistent object framework. This includes: o Deferred fetching o Simplified data storing o Reference counting o Caching for performance and structure preservation. All object inheriting from this class separate the request of a fetch from the actual database interaction. Fetches are deferred until data or services relating to the fetched object are accessed. This allows performance efficient access. In addition, this class supports a simplified model of data storage. The user of the class does not have to track whether the current object was already in the database or is a new object. The system will insert or update as appropriate. Derived classes of this class are not the actual objects requested by the user, but only handles to the objects. Thus when a user fetches a complex object out of the database, the user need not be concerned with memory management. All objects are reference counted. When references are no longer exist to the object, the object are deleted. This class maintains a cache of stored and fetched objects. If multiple clients fetch the same logical object from memory, they will be sharing it. This enables complex objects that share sub-components to be stored/fetched and maintain their structure. Protected View: Concrete and abstract subclasses must define the following operations: o An operator -> that returns the appropriate representation type. Concrete subclasses must define the following operations: o A NewRep() and NewRep(copyFrom) operation that returns new representation o A FetchedObjects() operation that returns access to the concrete cache. Private View: None

Attributes:

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
No Inheritance
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
No Inheritance
This is the current status.

Operations:

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
No Inheritance
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
No Inheritance
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
No Inheritance
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
No Inheritance
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
No Inheritance
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
No Inheritance
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
No Inheritance
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
No Inheritance
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
No Inheritance
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
No Inheritance
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
No Inheritance
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
No Inheritance
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
No Inheritance
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
No Inheritance
This operation compares the representations of this and another handle.

5.1.2.2 Class EcPoPersistentBase

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcPoHandle(Public) Coordinate

Description:

Public View: Not much of one. This class supports the concept of a persistent object having a unique ID. Protected View: This class provides the framework used by EcPoHandle to implement its deferred fetching capabilities. The methods for each subclass are specified here, but called by our Handle. Private View: None.

Attributes:

`myAmModified`

Privilege: Private
Data Type: EcTBoolean
Default Value: NOT IDENTIFIED
No Inheritance
This attribute contains a Boolean value. If TRUE, the object's data is changed and that it is set to modify so that the database will UPDATE and not INSERT.

`myFetchStatus`

Privilege: Private
Data Type: enum FetchStatus
Default Value: NOT IDENTIFIED
No Inheritance
This is the various states of fetching (FetchNotRequested, FetchRequested, FetchDone).

`myNumReferences`

Privilege: Public
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
No Inheritance
This is the counter of the number of handles accesses us.

`myObjectID`

Privilege: Private
Data Type: EcTPoDatabaseID
Default Value: NOT IDENTIFIED
No Inheritance
This attribute contains the current object ID.

`myReadOnlyStatus`

Privilege: Private
Data Type: EcTBoolean
Default Value: NOT IDENTIFIED
No Inheritance
This attribute constains the read only status which has a value of either TRUE or FALSE.

Operations:

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
No Inheritance
This operation returns a connection for the current object.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
No Inheritance
This operation returns the database for the current object.

`void EcPoPersistentBase (void)`

Privilege: Protected Operation
No Inheritance
Default constructor. Set reference count to 1 handle.

`void EcPoPersistentBase (copyFrom:EcPoPersistentBase&)`

Privilege: Protected Operation
No Inheritance
Copy constructor.

`EcUtStatus FetchPhaseII (dataFromSelector:RWDBReader&)`

Privilege: Protected Operation
No Inheritance
This operation processes the data in 'dataFromSelector' into the current object. The data in 'dataFromSelector' comes from executing the selector prepared in 'FetchPrep'.

```
 FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on&, foreignKey:const RWDBExpr&)
```

Privilege: Protection Not Identified
No Inheritance
This method shall process the selector, "dataToAcquire", to retreive all relevant data for this object that can be returned in a single database fetch. This method shall add any constraints to the "currentWhereClause". This method shall add a constraint that selects only the data that are identified by its primary key equalling "foreignKey".

```
const EcTPoDatabaseID& GetObjectID (void)
```

Privilege: Public
No Inheritance
This operation returns the unique object ID (primary key) for the current object. A valid ID determines whether we are currently in the database or not.

```
EcUtStatus InsertDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation will insert the current object into the database and set the current object's ID with a new unique value.

```
void PostFetchProcessing (status:EcUtStatus)
```

Privilege: Protected Operation
No Inheritance
This operation sets our internal state data after a fetch. Any derived class that does its own kind of fetching should also call it with the status from the Fetch.

```
void SetModified (value:EcTBoolean)
```

Privilege: Public
No Inheritance
This operation determines whether we are modified.

```
void SetObjectID (ID:const EcTPoDatabaseID&)
```

Privilege: Protected Operation
No Inheritance
This routine will set our object's ID. The ID determines whether the object is in the database or not.

`EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
No Inheritance
This method will update an existing object in the database that matches our current ID with the data in this object.

`const EcPoPersistentBase& operator= (assignFrom:const EcPoPersistentBase&)`

Privilege: Protected Operation
No Inheritance
This operation enables assignment. It is similar to copy.

`EcTBoolean operator== (equalTo:const EcPoPersistentBase&)`

Privilege: Public
No Inheritance
This operation returns a True value if IDs are the same.

`void ~EcPoPersistentBase (void)`

Privilege: Public
No Inheritance
Default destructor.

5.1.2.3 Class IoAdAdvertisement

Synopsis:

Parent Class: EcPoHandle
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcPoHandle

Description:

Public View: This is an abstract handle class for all advertisements. It provides the operator-> operation to access members of the base class IoAdAdvertisementRep. Protected View: We do not delete data in our dtor, our derived concrete class shall. We do not declare "NewRep()", our derived concrete class shall. We do not declare a "FetchedObjects" cache, our derived concrete class shall. Private View: None.

Attributes:

`myContact`

Privilege: Private
Data Type: IoAdContact
Default Value: NOT IDENTIFIED
No Inheritance
Who/What is responsible for the advertised entity.

`myCopyRight`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Any relevant copyright that applies to the entity being advertised.

`myDescription`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
No Inheritance
Long textual description of the advertised entity.

`myExpirationDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
No Inheritance
When am I no longer valid.

`myGroup`

Privilege: Private
Data Type: RWCString myGroup
Default Value: NOT IDENTIFIED
No Inheritance
Logical group I am part of for administration.

`myGuideURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
The Web URL for a guide to my entity.

`myStartDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
No Inheritance
When I am valid for advertisement.

`myTitle`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the unique description for me.

`myType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
No Inheritance
What derived type are we part of.

`myUR`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
No Inheritance
A universal reference to access my advertised entity.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

Operations:

`virtual RWDBConnection& Connection (void)`

Privilege: Protected Operation
No Inheritance
This operation returns a valid connection.

`virtual RWDBDatabase& Database (void)`

Privilege: Protected Operation
No Inheritance
This operation returns a valid database with data in it.

`static const char* DescriptionTableName (void)`

Privilege: Private
No Inheritance
This operation contain names to encapsulate database table.

```
virtual EcUtStatus FetchPhaseII
(dataFromSelector:RWDBReader&)
```

Privilege: Protected Operation
No Inheritance
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on, foreignKey:const RWDBExpr&)
```

Privilege: Protected Operation
No Inheritance
This operation prepares the selector to acquire all of classes data.

`IoAdContact GetContact (void)`

Privilege: Public
No Inheritance
This operation gets who/what is responsible for the advertised entity.

`const char* GetCopyRight (void)`

Privilege: Public
No Inheritance
This operation gets any relevant copyrights that apply to the entity being advertised.

`const char* GetDescription (void)`

Privilege: Public
No Inheritance
This operation gets a long textual description of the advertised entity.

`const char* GetGroup (void)`

Privilege: Public
No Inheritance
This operation gets the logical group I am part of for administration.

`const char* GetGuideURL (void)`

Privilege: Public
No Inheritance
This operation gets the Web URL for a guide to my entity.

`RWDBDateTime GetStartDate (void)`

Privilege: Public
No Inheritance
This operation gets the start date when the ad is valid.

`const char* GetTitle (void)`

Privilege: Public
No Inheritance
This operation gets a unique description of the title .

`AdvertisingType GetType (void)`

Privilege: Public
No Inheritance
This operation gets the derived type that the advertisement is part of.

`const char* GetUR (void)`

Privilege: Public
No Inheritance
This operation gets the universal reference to access the advertised entity.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
No Inheritance
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid.

`void IoAdAdvertisementRep (type:AdvertisingType)`

Privilege: Protected Operation
No Inheritance
Default constructor.

`void IoAdAdvertisementRep (copyFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
No Inheritance
Copy constructor.

`EcTBoolean IsValid (void)`

Privilege: Protected Operation
No Inheritance
This operation checks the contents of the object for consistency and data rules: o Contact is valid o Title, Description, and Group are not NULL

`static const char* MasterTableName (void)`

Privilege: Private
No Inheritance
This operation contain names to encapsulate database table.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
No Inheritance
This operation writes contents to stream.

`void SetContact (value:IoAdContact&)`

Privilege: Public
No Inheritance
This operation sets who/what is responsible for the advertised entity.

`void SetCopyRight (value:const char*)`

Privilege: Public
No Inheritance
This operation sets any relevant copyrights that apply to the entity being advertised.

`void SetDescription (value:const char*)`

Privilege: Public
No Inheritance
This operation sets the long textual description of the advertised entity.

`void SetGroup (value:const char*)`

Privilege: Public
No Inheritance
This operation sets the logical group I am part of for administration.

`void SetGuideURL (value:const char*)`

Privilege: Public
No Inheritance
This operation sets the Web URL for a guide to my entity.

`void SetStartDate (value:RWDBDateTime)`

Privilege: Public
No Inheritance
This operation sets the start date when the ad is valid.

`void SetTitle (value:const char*)`

Privilege: Public
No Inheritance
This operation sets a unique description of the title.

`void SetUR (value:const char)`

Privilege: Public
No Inheritance
This operation sets the universal reference to access the advertised entity.

```
void SetValues (type:AdvertisingType,title:const
char*,description:const char*, guideURL:const
char*,group:const char*,UR:const char*,copyRight:const
char*, contact:IoAdContact&)
```

Privilege: Protected Operation
No Inheritance
This operation sets values for the parameters passed in the argument lists.

`static const char* URTableName (void)`

Privilege: Private
No Inheritance
This operation contains names to encapsulate database table.

`virtual EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID.

`IoAdAdvertisementRep* operator-> (void)`

Privilege: Public
No Inheritance
This operation enables the -> operation to access members of the base class IoAdAdvertisementRep.

`const IoAdAdvertisementRep& operator= (assignFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
No Inheritance
This operator enables the assignment of two objects.

`void ~IoAdAdvertisement (void)`

Privilege: Public
No Inheritance
Default destructor.

`void ~IoAdAdvertisementRep (void)`

Privilege: Public
No Inheritance
Default destructor.

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

```
EcPoPersistentBase* NewRep (copyFrom: const
EcPoPersistentBase*)
```

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

```
void Store (void)
```

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

```
EcPoPersistentBase* operator-> (void)
```

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

```
const EcPoHandle& operator= (assignFrom: EcPoHandle&)
```

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

```
EcTBoolean operator== (equalTo: EcPoHandle&)
```

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

5.1.2.4 Class IoAdContact

Synopsis:

Parent Class: EcPoHandle
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcPoHandle

Description:

Public View   This class represents a person or organization responsible for an advertisement. It basically holds data. It supports the operations of the persistent object framework.   Data and Services should not be accessed directly, but through IoAdContact. IoAdContact performs handle services. Because fetching is deferred, any access can generate a database error. Therefore, one should check the handles status when appropriate. Protected View: None. Private View: None.

Attributes:

`myCityName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the city name of the address of the contact person.

`myCountyName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the country name of the address of the contact person.

`myEMail`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the e-mail address of the contact person.

`myFaxNumber`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the fax phone number of the contact person.

`myFirstName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the first name of the contact person.

`myLastName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the last name of the contact person.

`myMiddleName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the middle name of the contact person.

`myOrgName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the name of the organization of the contact person.

`myStateName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the state of the address of the contact person

`myStreetName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the street name of the address of the contact person.

`myTelNumber`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the telephone number of the contact person.

`myZipCode`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the zip code of the address of the contact person.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

Operations:

`virtual RWDBConnection& Connection (void)`

Privilege: Protected Operation
No Inheritance
This operation returns the connection for our class.

`static const char* ContactTableName (void)`

Privilege: Private
No Inheritance
This operation returns contact database table name.

`virtual RWDBDatabase& Database (void)`

Privilege: Protected Operation
No Inheritance
This operation returns the database for our class.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
No Inheritance
This routine selects an object from the database based on its state. For contact, the unique application state is FirstName, LastName, and Org.

```
virtual EcUtStatus FetchPhaseII (dataFromSelector:
RWDBReader&)
```

Privilege: Public
No Inheritance
This operation gets all the data from FetchPrep.

```
 FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on&, foreignKey:const RWDBExpr&)
```

Privilege: Protection Not Identified
No Inheritance
This operation prepares the selector to acquire all of classes data.

```
const char* GetCityName (void)
```

Privilege: Public
No Inheritance
This operation returns the city name of the address of the contact person.

```
const char* GetCountryName (void)
```

Privilege: Public
No Inheritance
This operation returns the country name of the address of the contact person.

```
const char* GetEMail (void)
```

Privilege: Public
No Inheritance
This operation returns the e-mail address of the contact person.

```
const char* GetFaxNumber (void)
```

Privilege: Public
No Inheritance
This operation returns the fax phone number of the contact person.

```
const char* GetFirstName (void)
```

Privilege: Public
No Inheritance
This operation returns the first name of the contact person.

`const char* GetLastName (void)`

Privilege: Public
No Inheritance
This operation returns the last name of the contact person.

`const char* GetMiddleName (void)`

Privilege: Public
No Inheritance
This operation returns the middle name of the contact person.

`const char* GetOrgName (void)`

Privilege: Public
No Inheritance
This operation returns the name of the organization of the contact person.

`const char* GetStateName (void)`

Privilege: Public
No Inheritance
This operation returns the state name of the address of the contact person.

`const char* GetStreetName (void)`

Privilege: Public
No Inheritance
This operation returns the street name of the address of the contact person.

`const char* GetTelNumber (void)`

Privilege: Public
No Inheritance
This operation returns the phone number of the contact person.

`const char* GetZipCode (void)`

Privilege: Public
No Inheritance
This operation returns the zip code of the address of the contact person.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
No Inheritance
This operation stores all attributes in the database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid.

`void IoAdContactRep (void)`

Privilege: Public
No Inheritance
Default constructor.

`void IoAdContactRep (copyFrom:const IoAdContactRep)`

Privilege: Public
No Inheritance
Copy constructor.

`virtual EcTBoolean IsValid (void)`

Privilege: Public
No Inheritance
This operation determines whether the data meets the requirements.

`virtual void PrintMembers (out: ostream&)`

Privilege: Public
No Inheritance
This operation prints contents to stream.

`void SetCityName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the city name of the address of the contact person.

`void SetCountryName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the country name of the address of the contact person.

`void SetEMail (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the e-mail address of the contact person.

`void SetFaxNumber (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the fax phone number of the contact person.

`void SetFirstName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the first name of the contact person.

`void SetLastName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the last name of the contact person.

`void SetMiddleName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the middle name of the contact person.

`void SetOrgName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the name of the organization of the contact person.

`void SetStateName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the state name of the address of the contact person.

`void SetStreetName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the street name of the address of the contact person.

`void SetTelNumber (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the phone number of the contact person.

```
void SetValues (lastName:const char*,middleName:const
char*,firstName:const char*, streetName:co nst
char*,cityName:const char*,stateName:const char*,
countryName:const char*,zi pCode:const char*,eMail:const
char*, telNaumber:const char*,faxNumber:const char
*,orgName:const char*)
```

Privilege: Public
No Inheritance
This operation sets values for the parameters passed in the argument lists.

```
void SetZipCode (value: const char*)
```

Privilege: Public
No Inheritance
This operation sets the zip code of the address of the contact person.

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID.

```
const IoAdContactRep& operator= (asignFrom: const
IoAdContactRep)
```

Privilege: Public
No Inheritance
This operation enables the assignment of two objects.

```
void ~IoAdContactRep (void)
```

Privilege: Public
No Inheritance
Default destructor.

```
void Clone (cloneFrom: EcPoHandle&)
```

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

5.1.2.5 Class IoAdContactSearchCommand

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

Public View: This class provides interfaces for applications to search the set of all derived classes of advertisement contacts by specifying criterion. The persistent data will be stored into a result list for additional searches or access. Users should set up options of how to search (filtering, patterns, how many results to return) and then call the search interfaces. While concrete, derived objects are placed in the results list, we return a list of advertisement contact objects. If one wants to access members of the advertisement contact, use the associated Contact-List class to see the Advertisement Contacts. Protected View: None. Private View: None.

Attributes:

`myContactTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
No Inheritance
This is an object to monitor the physical database table.

`myMatchType`

Privilege: Private
Data Type: MatchTypeEnum
Default Value: NOT IDENTIFIED
No Inheritance
This is the match type (Prefix/Contain/Exact) that the pattern will be compared.

`myPattern`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
This is the matched pattern to be searched.

`myResults`

Privilege: Private
Data Type: IoAdContactList
Default Value: NOT IDENTIFIED
No Inheritance
This attribute contains results of found advertisement contacts.

`mySearchLimit`

Privilege: Private
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
No Inheritance
This is the search limit for the match type (Prefix/Contain/Exact) in which the pattern will be compared.

Operations:

`EcTVoid ClearResults (void)`

Privilege: Public
No Inheritance
This operation clears all found advertisement contacts.

`EcUtStatus CommonQueryProcessing (&select:RWDBSelectorBase)`

Privilege: Private
No Inheritance
This operation processes the search of persistent object list for pattern matched.

`RWDBSelector CommonSelector (expr:RWDBCriterion)`

Privilege: Private
No Inheritance
This operation constructs the search attribute for persistent objects.

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
No Inheritance
This operation returns the default database connection.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
No Inheritance
This operation connects to the database server.

`RWDBCriterion Expr (curTable:RWDBTable, curColumn:RWDBColumn)`

Privilege: Private
No Inheritance
This operation formulates the search criterion according to the matched pattern.

`RWDBCriterion ExprByFirstName (void)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for First Name Search.

`RWDBCriterion ExprByLastName (void)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for Last Name Search.

`RWDBCriterion ExprByOrganization (void)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for Organization Name search.

`RWCString GetPattern (void)`

Privilege: Private
No Inheritance
This operation retrieves the matched pattern to be searched.

`const IoAdContactList& GetResults (void)`

Privilege: Public
No Inheritance
This operation returns the found advertisement contacts.

`EcUtStatus InsertContactIntoResults (id:EcTPoDatabaseID)`

Privilege: Private
No Inheritance
This operation creates an advertisement with the characteristic specified.

`void IoAdContactSearchCommand (void)`

Privilege: Public
No Inheritance
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
No Inheritance
This operation resets all the search criterion to default. Match types defaulted to prefix, and default search limit returns all.

`EcUtStatus SearchByFirstName (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for any Advertisement Contact that contains a substring of matchTo in its First Name and appends found contact to the current results set.

`EcUtStatus SearchByLastName (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for any Advertisement Contact that contains a substring of matchTo in its Last Name and appends found contact to the current results set.

`EcUtStatus SearchByOrganization (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for any Advertisement Contact that contains a substring of matchTo in its Organization and append found contact to the current results set.

`EcUtStatus SearchByText (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for any Contact that contains a substring of matchTo in its Organization, Last/First Name and append found Advertisement Contact to the current results set.

`EcTVoid SetMatchType (matchType:MatchTypeEnum)`

Privilege: Public
No Inheritance
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared.

`EcTVoid SetPattern (matchTo:RWCString)`

Privilege: Private
No Inheritance
This operation stores the matched pattern to be searched.

`EcTVoid SetSearchLimit (maxToReturn:EcTUInt)`

Privilege: Public
No Inheritance
This operation enables the specification of the maximum number of results to return for a given search. If not specified, all that match will be returned.

`void ~IoAdContactSearchCommand (void)`

Privilege: Public
No Inheritance
Default destructor.

5.1.2.6 Class IoAdMimeServiceAdv

Synopsis:

Parent Class: IoAdService
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdService

Description:

Public View: This entity class supports operations to allow the definition, storage, and retrieval of an advertisement of a mime service. Member data and operations should not be accessed directly, but through IoAdMimeServiceAdv. IoAdMimeServiceAdv performs handle services. Because fetching is deferred, any access can generate a database error. Therefore, one should check the handle status when appropriate. Unless specified, the string values can be empty. Protected View: We inherit our Database() and Connection() from IoAdAdvertisement. Private View: None.

Attributes:

`myMimeType`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the name of the Mime type of the the service.

`myMineURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the URL of the Mime type of the service to be accessed.

`myContact`

Privilege: Private
Data Type: IoAdContact
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Who/What is responsible for the advertised entity.

`myCopyRight`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Any relevant copyright that applies to the entity being advertised.

`myDescription`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Long textual description of the advertised entity.

`myExpirationDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When am I no longer valid.

`myGroup`

Privilege: Private
Data Type: RWCString myGroup
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Logical group I am part of for administration.

`myGuideURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
The Web URL for a guide to my entity.

`myStartDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When I am valid for advertisement.

`myTitle`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Stores the unique description for me.

`myType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
What derived type are we part of.

`myUR`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
A universal reference to access my advertised entity.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

`myProducts`

Privilege: Private
Data Type: IoAdProductReferenceList
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
What products can we apply to.

`myProvider`

Privilege: Private
Data Type: IoAdProvider
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Who is providing this product.

`myServiceClass`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Stores the name of the service class for the advertised service.

`myServiceTypeId`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Stores the signature of the service.

`myServideName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Stores the name of the advertised service.

Operations:

`EcUtStatus DeleteBasicData (void)`

Privilege: Private
No Inheritance
This operation deletes all attributes in database.

`virtual EcUtStatus DeleteDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation deletes this object in the database for its current object ID.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
No Inheritance
This routine selects an object from database based on its state. For contact, the unique application state is MimeTypeID, MimeURL and ServiceID.

```
virtual EcUtStatus FetchPhaseII (dataFrom Selector :
RWDBReader&)
```

Privilege: Public
No Inheritance
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on&, foreignKey:RWDBExpr&)
```

Privilege: Public
No Inheritance
This operation prepares the selector to acquire all of classes data.

`const char* GetMimeType (void)`

Privilege: Public
No Inheritance
This operation returns the name of the Mime type of the service.

`const char* GetMimeURL (void)`

Privilege: Public
No Inheritance
This operation returns the URL of the Mime type of the service.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
No Inheritance
This operation stores all attributes in database.

```
virtual EcUtStatus InsertDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

```
void IoAdMimeServiceAdvRep (copyFrom :
IoAdMimeServiceAdvRep&)
```

Privilege: Public
No Inheritance
Copy constructor.

```
void IoAdMimeServiceRep (void)
```

Privilege: Public
No Inheritance
Default constructor.

```
virtual EcTBoolean IsValid (void)
```

Privilege: Public
No Inheritance
This operation checks the contents of the object for consistency and data rules: o Provider is valid   o Ad Base is valid   o MimeType and MimeURL are not NULL.

```
const char* MimeServiceAdvTableName (void)
```

Privilege: Private
No Inheritance
This operation gets the Mime Service Advertisement database Table Name.

```
virtual void PrintMembers (out : ostream&)
```

Privilege: Public
No Inheritance
This operation writes contents to stream.

```
void SetMimeType (value:const char*)
```

Privilege: Public
No Inheritance
This operation is used to set a new Mime type name for the service.

`void SetMimeURL (value: const char*)`

Privilege: Public
No Inheritance
This operation is used to set a new URL of the Mime type of the service.

```
void SetValues (title:const char*,description:const
char*,guideURL:const char*, group:const
char*,copyRight:const char*,contact:IoAdContact,
provider:IoAdProvider,mimeType:const char*, mimeURL:const
char*)
```

Privilege: Public
No Inheritance
This operation sets values for the parameters passed in the argument lists.

`EcUtStatus UpdateBasicData (void)`

Privilege: Private
No Inheritance
This operation stores all attributes in database.

`virtual EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID.

```
const IoAdMimeServiceAdvRep& operator= (assignFrom :
IoAdMimeServiceAdvRep&)
```

Privilege: Public
No Inheritance
This operation enables the assignment of two objects.

`void ~IoAdMimeServiceAdvRep (void)`

Privilege: Public
No Inheritance
Default destructor.

`virtual RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid connection.

`virtual RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid database with data in it.

`static const char* DescriptionTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

```
virtual EcUtStatus FetchPhaseII
(dataFromSelector:RWDBReader&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on, foreignKey:const RWDBExpr&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation prepares the selector to acquire all of classes data.

`IoAdContact GetContact (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets who/what is responsible for the advertised entity.

`const char* GetCopyRight (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets any relevant copyrights that apply to the entity being advertised.

`const char* GetDescription (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a long textual description of the advertised entity.

`const char* GetGroup (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the logical group I am part of for administration.

`const char* GetGuideURL (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the Web URL for a guide to my entity.

`RWDBDateTime GetStartDate (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the start date when the ad is valid.

`const char* GetTitle (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a unique description of the title .

`AdvertisingType GetType (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the derived type that the advertisement is part of.

`const char* GetUR (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the universal reference to access the advertised entity.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation inserts our data if it is logically valid.

`void IoAdAdvertisementRep (type:AdvertisingType)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Default constructor.

`void IoAdAdvertisementRep (copyFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Copy constructor.

`EcTBoolean IsValid (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation checks the contents of the object for consistency and data rules: o Contact is valid o Title, Description, and Group are not NULL

`static const char* MasterTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation writes contents to stream.

`void SetContact (value:IoAdContact&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets who/what is responsible for the advertised entity.

`void SetCopyRight (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets any relevant copyrights that apply to the entity being advertised.

`void SetDescription (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the long textual description of the advertised entity.

```
void SetGroup (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the logical group I am part of for administration.

```
void SetGuideURL (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the Web URL for a guide to my entity.

```
void SetStartDate (value:RWDBDateTime)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the start date when the ad is valid.

```
void SetTitle (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets a unique description of the title.

```
void SetUR (value:const char)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the universal reference to access the advertised entity.

```
void SetValues (type:AdvertisingType,title:const
char*,description:const char*, guideURL:const
char*,group:const char*,UR:const char*,copyRight:const
char*, contact:IoAdContact&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation sets values for the parameters passed in the argument lists.

```
static const char* URTableName (void)
```

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contains names to encapsulate database table.

`virtual EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation updates this object in the database for its current object ID.

`IoAdAdvertisementRep* operator-> (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation enables the -> operation to access members of the base class IoAdAdvertisementRep.

`const IoAdAdvertisementRep& operator= (assignFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operator enables the assignment of two objects.

`void ~IoAdAdvertisement (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void ~IoAdAdvertisementRep (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

`EcUtStatus AddProduct (productToAdd:IoAdProduct)`

Privilege: Public
Inherited From: IoAdService
This operation defines a new product applies to this service. This in turn also defines this service is applicable to the given product.

`EcUtStatus DeleteProduct (productToDelete:IoAdProduct)`

Privilege: Public
Inherited From: IoAdService
This operation removes a defined product from this service. This in turn also removes this service from the product.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
Inherited From: IoAdService
This routine selects an object from the database based on its state. For contact, the unique applicable state is ServiceClassID, ServiceName, and ServiceID

`virtual EcUtStatus FetchPhaseII (dataFromSelector:RWDBReader&)`

Privilege: Public
Inherited From: IoAdService
This operation gets all the data from FetchPrep.

`FetchPrep (dataToAcquire:RWDBSelector&, currentWhereClause:RWDBCriterion&, foreignKey:const RWDBExpr&)`

Privilege: Protection Not Identified
Inherited From: IoAdService
This operation prepares the selector to acquire all of classes data.

`IoAdProvider GetProvider (void)`

Privilege: Public
Inherited From: IoAdService
This operation gets the provider who is providing the product.

`const char* GetServiceClass (void)`

Privilege: Public
Inherited From: IoAdService
This operation returns the current service class name of the advertised service.

`const char* GetServiceName (void)`

Privilege: Public
Inherited From: IoAdService
This operation returns the name of the advertised service.

`EcTInt GetServiceTypeId (void)`

Privilege: Public
Inherited From: IoAdService
This operation defines the signature of the service.

`EcUtStatus InsertBasicData (void)`

Privilege: Protected Operation
Inherited From: IoAdService
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdService
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

`void IoAdServiceRep (void)`

Privilege: Public
Inherited From: IoAdService
This constructor sets our advertising type. It initializes our list of products to know we are the source object.

`void IoAdServiceRep (copyFrom:IoAdServiceRep&)`

Privilege: Public
Inherited From: IoAdService
Copy constructor.

`virtual EcTBoolean IsValid (void)`

Privilege: Public
Inherited From: IoAdService
This operation checks the contents of the object for consistency and data rules. o Provider is valid o Ad Base is valid o Service class and service name are not NULL

`EcTUInt NumberOfProducts (void)`

Privilege: Public
Inherited From: IoAdService
This operation defines the number of products that service applies to.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
Inherited From: IoAdService
This operation writes contents to stream.

`IoAdProductReferenceListIter ProductIter (void)`

Privilege: Public
Inherited From: IoAdService
This operation provides a RWTPSlist-like iterator for all applicable products.

`const char* ServiceTableName (void)`

Privilege: Private
Inherited From: IoAdService
This is a private function to encapsulate database table.

`void SetProvider (value:IoAdProvider)`

Privilege: Public
Inherited From: IoAdService
This operation sets the myProvider attribute.

`void SetServiceClass (value:const char*)`

Privilege: Public
Inherited From: IoAdService
This operation is used to set a new service class name for the advertised service.

`void SetServiceName (value:const char*)`

Privilege: Public
Inherited From: IoAdService
This operation is used to set the name of the advertised service.

`void SetServiceTypeId (value:EcTInt)`

Privilege: Public
Inherited From: IoAdService
This operation sets the signature of the service.

```
 SetValues (title: const char*, description: const char*,
guideURL const char*, group: cons t char*, ur:const char*,
copyRight:const char*, contact: IoAdContact, provider:
IoAdProvider, serviceClass: const char*, serviceName:const
char*, serviceType Id: EcTInt)
```

Privilege: Protection Not Identified
Inherited From: IoAdService

`virtual EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdService
This operation updates this object in the database for its current object ID. This routine is part of the persistent framework.

`const IoAdServiceRep& operator= (assignFrom:IoAdServiceRep&)`

Privilege: Public
Inherited From: IoAdService
This operator enables the assignment of two objects.

`void ~IoAdServiceRep (void)`

Privilege: Public
Inherited From: IoAdService
Default destructor.

5.1.2.7 Class IoAdProduct

Synopsis:

Parent Class: IoAdAdvertisement
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdAdvertisement

Description:

Public View: This entity class supports operations to allow the definition, storage and retrieval of a advertisement of a data product. Typically, this product is an ECS collection. It can also be other kinds of collections or other general data. Member data and functions should not be accessed directly, but through IoAdProduct. IoAdProduct performs handle services. Because fetching is deferred, any access can generate a database error. Therefore, one should check the handles status when appropriate. Unless specified, the string values can be empty. Protected View: We inherit our Database() and Connection() from IoAdAdvertisement. Private View: None.

Attributes:

`myContact`

Privilege: Private
Data Type: IoAdContact
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Who/What is responsible for the advertised entity.

`myCopyRight`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Any relevant copyright that applies to the entity being advertised.

`myDescription`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Long textual description of the advertised entity.

`myExpirationDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When am I no longer valid.

`myGroup`

Privilege: Private
Data Type: RWCString myGroup
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Logical group I am part of for administration.

`myGuideURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
The Web URL for a guide to my entity.

`myStartDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When I am valid for advertisement.

`myTitle`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Stores the unique description for me.

`myType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
What derived type are we part of.

`myUR`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
A universal reference to access my advertised entity.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

`myCollectionID`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the ID of the collection.

`myProductTypeName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the product name.

`myProvider`

Privilege: Private
Data Type: IoAdProvider
Default Value: NOT IDENTIFIED
No Inheritance
Who is providing this product.

`myServices`

Privilege: Private
Data Type: IoAdServiceReferenceList
Default Value: NOT IDENTIFIED
No Inheritance
What services can be applied to this data.

Operations:

`virtual RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid connection.

`virtual RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid database with data in it.

`static const char* DescriptionTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

`virtual EcUtStatus FetchPhaseII (dataFromSelector:RWDBReader&)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on, foreignKey:const RWDBExpr&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation prepares the selector to acquire all of classes data.

```
IoAdContact GetContact (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets who/what is responsible for the advertised entity.

```
const char* GetCopyRight (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets any relevant copyrights that apply to the entity being advertised.

```
const char* GetDescription (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a long textual description of the advertised entity.

```
const char* GetGroup (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the logical group I am part of for administration.

```
const char* GetGuideURL (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the Web URL for a guide to my entity.

```
RWDBDateTime GetStartDate (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the start date when the ad is valid.

```
const char* GetTitle (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a unique description of the title .

`AdvertisingType GetType (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the derived type that the advertisement is part of.

`const char* GetUR (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the universal reference to access the advertised entity.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation inserts our data if it is logically valid.

`void IoAdAdvertisementRep (type:AdvertisingType)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Default constructor.

`void IoAdAdvertisementRep (copyFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Copy constructor.

`EcTBoolean IsValid (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation checks the contents of the object for consistency and data rules: o Contact is valid   o Title, Description, and Group are not NULL

`static const char* MasterTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation writes contents to stream.

`void SetContact (value:IoAdContact&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets who/what is responsible for the advertised entity.

`void SetCopyRight (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets any relevant copyrights that apply to the entity being advertised.

`void SetDescription (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the long textual description of the advertised entity.

`void SetGroup (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the logical group I am part of for administration.

`void SetGuideURL (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the Web URL for a guide to my entity.

`void SetStartDate (value:RWDBDateTime)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the start date when the ad is valid.

`void SetTitle (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets a unique description of the title.

```
void SetUR (value:const char)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the universal reference to access the advertised entity.

```
void SetValues (type:AdvertisingType,title:const
char*,description:const char*, guideURL:const
char*,group:const char*,UR:const char*,copyRight:const
char*, contact:IoAdContact&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation sets values for the parameters passed in the argument lists.

```
static const char* URTableName (void)
```

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contains names to encapsulate database table.

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation updates this object in the database for its current object ID.

```
IoAdAdvertisementRep* operator-> (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation enables the -> operation to access members of the base class IoAdAdvertisementRep.

```
const IoAdAdvertisementRep& operator=
(assignFrom:IoAdAdvertisementRep&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operator enables the assignment of two objects.

```
void ~IoAdAdvertisement (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void ~IoAdAdvertisementRep (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

```
void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly =
EcDFalse: EcTBoolean)
```

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

```
void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly =
EcDFalse: EcTBoolean)
```

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

`EcUtStatus AddService (serviceToAdd:IoAdService)`

Privilege: Public
No Inheritance
This operation defines a new service to apply to this product

`EcUtStatus DeleteService (serviceToDelete: IoAdService)`

Privilege: Public
No Inheritance
This operation removes an existing service that applied to this product.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
No Inheritance
This routine selects an object from the database based on its state. For contact, the unique application is ProductName and CollectionID

`virtual EcUtStatus FetchPhaseII (dataFromSelector: RWDBReader&)`

Privilege: Public
No Inheritance
This operation gets all the data from FetchPrep. This routine is part of the persistent framework.

`FetchPrep (dataToAcquire: RWDBSelector&, currentWhereClause: RWDBCriterion&, fo reignKey: const RWDBExpr&)`

Privilege: Protection Not Identified
No Inheritance

`const char* GetCollectionID (void)`

Privilege: Public
No Inheritance
This operation gets the ID of the collection.

`const char* GetProductTypeName (void)`

Privilege: Public
No Inheritance
This operation returns the product type name.

`IoAdProvider GetProvider (void)`

Privilege: Public
No Inheritance
This operation gets the provider who is providing the product.

`EcUtStatus InsertBasicData (void)`

Privilege: Protected Operation
No Inheritance
This operation stores all attributes in database.

`virtual EcutStatus InsertDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

`void IoAdProductRep (copyFrom : IoAdProductRep&)`

Privilege: Public
No Inheritance
Copy constructor.

`void IoAdProductRep (void)`

Privilege: Public
No Inheritance
Default constructor.

`virtual EctBoolean IsValid (void)`

Privilege: Public
No Inheritance
This operation checks the contents of the object for consistency and data rules: o Provider is valid   o Ad Base is valid   o Collection ID is not NULL

`EcTUint NumberOfServices (void)`

Privilege: Public
No Inheritance
 This operation defines how many services apply to this product.

`virtual void PrintMembers (out: ostream&)`

Privilege: Public
No Inheritance
This operation writes contents to stream.

```
const char* ProductTableName (void)
```

Privilege: Private
No Inheritance
This is a private function to encapsulate table name.

```
IoAdServiceReferenceListIter ServiceIter (void)
```

Privilege: Public
No Inheritance
This operation provides a RWTPSlist-like iterator for all defined services

```
void SetCollectionID (value: const char*)
```

Privilege: Public
No Inheritance
This operation sets the ID of the collection.

```
void SetProductTypeName (value: const char*)
```

Privilege: Public
No Inheritance
This operation is used to set a new product type name.

```
void SetProvider (value: IoAdProvider)
```

Privilege: Public
No Inheritance
This operation sets the myProvider attribute.

```
 SetValues (title:const char*, description:const char*,
guideURL:const char*, group:const c har*, UR:const char*,
copyRight:const char*, contact:IoAdContact, provider:IoAdP
rovider, productTypeName:const char*, CollectionID:const
char*)
```

Privilege: Protection Not Identified
No Inheritance

```
virtual UpdateDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID. This routine is part of the persistent framework.

```
const IoAdProductRep& operator= (assignFrom :
IoAdProductRep&)
```

Privilege: Public
No Inheritance
This operator enables the assignment of two objects.

```
void ~IoAdProductRep (void)
```

Privilege: Public
No Inheritance
Default destructor.

5.1.2.8 Class IoAdProductSearchCommand

Synopsis:

Parent Class: IoAdSearchCommand
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdSearchCommand

Description:

Public View: This class provides interfaces for applications to search the set of product advertisements by specifying options and criterion. The persistent data will be stored into a results list for additional searches or access. Users should set up options of how to search (filtering, patterns, how many results to return) and then call the search interfaces. Protected View: None. Private View: None.

Attributes:

```
myResults
```

Privilege: Private
Data Type: IoAdProductList
Default Value: NOT IDENTIFIED
No Inheritance
Contains the matched Product Advertisement to user.

```
myAdvType
```

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
What kind of search type (product, service, provider).

`myDescTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the description table of the database.

`myMasterTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the master table.

`myMatchType`

Privilege: Private
Data Type: MatchTypeEnum
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains types of pattern (Prefix/Contain/Exact) matching allowable on string searches.

`myPattern`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains the matched pattern to be searched.

`myResults`

Privilege: Private
Data Type: IoAdAdvertisementList
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains accummulated set of results for this search.

`mySearchLimit`

Privilege: Private
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
This is the match type (Prefix/Contain/Exact) that the pattern will be compared, or else all the matched will be found.

Operations:

`EcTVoid ClearResults (void)`

Privilege: Public
No Inheritance
This operation clears the matched object linked list.

`const IoAdProductList& GetResults (void)`

Privilege: Public
No Inheritance
This operation returns the matched Product Advertisement to user.

`void IoAdProductSearchCommand (void)`

Privilege: Public
No Inheritance
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
No Inheritance
This operation resets all the search criterion to default. Default Adv type searches Product Advertisement type.

`EcUtStatus SearchByCollectionID (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for Product Advertisement that contains a substring of matchTo in its Collection ID and appends found Product advertisements to the current results set.

`EcTVoid SetSearchType (typeFilter: AdvertisingType)`

Privilege: Protected Operation
No Inheritance
This operation sets the type of search.

`void ~IoAdProductSearchCommand (void)`

Privilege: Public
No Inheritance
Default destructor.

`EcTVoid ClearResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation clears the matched object linked list.

`RWDBCriterion CommonExpr (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Adv Type search.

`EcUtStatus CommonQueryProcessing (&select:RWDBSelectorBase)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation processes the search of persistent object list for pattern matched.

`RWDBSelector CommonSelector (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation constructs the search attribute for persistent objects.

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation returns the default database connection.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation connects to the database server.

`RWDBCriterion Expr (curTable:RWDBTable, curColumn:RWDBColumn)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion according to the matched pattern.

`RWDBCriterion ExprByDescription (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Description search.

`RWDBCriterion ExprByTitle (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Title search.

`RWDBCriterion ExprByURL (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for URL search.

`RWCString GetPattern (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation retrieves the matched pattern to be searched.

`const IoAdAdvertisementList& GetResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation returns the matched Advertisement object to user.

```
EcUtStatus InsertAdvertisementIntoResults
(id:EcTPDatabaseID, type: AdvertisingType)
```

Privilege: Private
Inherited From: IoAdSearchCommand
This operation creates an advertisement with the characteristic specified.

`void IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation resets all the search criterion to default. Default Adv type searches all product/provider/service type, match Type is defaulted to prefix, and default search limit returns all.

`EcUtStatus SearchByText (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title, URL, or Description and appends found advertisements to the current results set.

`EcUtStatus SearchByTitle (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title and appends found advertisements to the current results set.

`EcTVoid SetMatchType (matchType:MatchTypeEnum)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared.

`EcTVoid SetPattern (matchTo:RWCString)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation stores the matched pattern to be searched.

`EcTVoid SetSearchLimit (maxToReturn:EcTUInt)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared or else all the matched will be found.

`EcTVoid SetSearchType (typeFilter:AdvertisingType)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets all the Adv Type desired, if not specified, all Adv Type will be returned.

`void ~IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default destructor.

5.1.2.9 Class IoAdProvider

Synopsis:

Parent Class: IoAdAdvertisement
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdAdvertisement

Description:

Public View: This entity class supports operations to allow the definition, storage and retrieval of a advertisement of a data/service provider. Typically, this provider is a DAAC. It can also be any other organization that provides data or services. Member data and functions should not be accessed directly, but through IoAdProvider. IoAdProvider performs handle services. Because fetching is deferred, any access can generate a database error. Therefore, one should check the handles status when appropriate. Unless specified, the string values can be empty. Protected View: We inherit our Database() and Connection() from IoAdAdvertisement. Private View: None.

Attributes:

`myContact`

Privilege: Private
Data Type: IoAdContact
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Who/What is responsible for the advertised entity.

`myCopyRight`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Any relevant copyright that applies to the entity being advertised.

`myDescription`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Long textual description of the advertised entity.

`myExpirationDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When am I no longer valid.

`myGroup`

Privilege: Private
Data Type: RWCString myGroup
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Logical group I am part of for administration.

`myGuideURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
The Web URL for a guide to my entity.

`myStartDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When I am valid for advertisement.

`myTitle`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Stores the unique description for me.

`myType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
What derived type are we part of.

`myUR`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
A universal reference to access my advertised entity.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

`myAccessRestriction`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores a text description of any access restrictions imposed by the provider.

`myOrganizationName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the organization name of the provider.

`myProviderRole`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the name of the role performed by the provider of the advertisement.

Operations:

`virtual RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid connection.

`virtual RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid database with data in it.

`static const char* DescriptionTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

```
virtual EcUtStatus FetchPhaseII
(dataFromSelector:RWDBReader&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on, foreignKey:const RWDBExpr&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation prepares the selector to acquire all of classes data.

`IoAdContact GetContact (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets who/what is responsible for the advertised entity.

`const char* GetCopyRight (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets any relevant copyrights that apply to the entity being advertised.

`const char* GetDescription (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a long textual description of the advertised entity.

`const char* GetGroup (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the logical group I am part of for administration.

`const char* GetGuideURL (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the Web URL for a guide to my entity.

`RWDBDateTime GetStartDate (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the start date when the ad is valid.

`const char* GetTitle (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a unique description of the title .

`AdvertisingType GetType (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the derived type that the advertisement is part of.

`const char* GetUR (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the universal reference to access the advertised entity.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation inserts our data if it is logically valid.

`void IoAdAdvertisementRep (type:AdvertisingType)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Default constructor.

`void IoAdAdvertisementRep (copyFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Copy constructor.

`EcTBoolean IsValid (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation checks the contents of the object for consistency and data rules: o Contact is valid o Title, Description, and Group are not NULL

`static const char* MasterTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation writes contents to stream.

`void SetContact (value:IoAdContact&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets who/what is responsible for the advertised entity.

`void SetCopyRight (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets any relevant copyrights that apply to the entity being advertised.

`void SetDescription (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the long textual description of the advertised entity.

`void SetGroup (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the logical group I am part of for administration.

`void SetGuideURL (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the Web URL for a guide to my entity.

`void SetStartDate (value:RWDBDateTime)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the start date when the ad is valid.

`void SetTitle (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets a unique description of the title.

`void SetUR (value:const char)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the universal reference to access the advertised entity.

```
void SetValues (type:AdvertisingType,title:const
char*,description:const char*, guideURL:const
char*,group:const char*,UR:const char*,copyRight:const
char*, contact:IoAdContact&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation sets values for the parameters passed in the argument lists.

`static const char* URTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contains names to encapsulate database table.

`virtual EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation updates this object in the database for its current object ID.

`IoAdAdvertisementRep* operator-> (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation enables the -> operation to access members of the base class IoAdAdvertisementRep.

```
const IoAdAdvertisementRep& operator=
(assignFrom:IoAdAdvertisementRep&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operator enables the assignment of two objects.

`void ~IoAdAdvertisement (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void ~IoAdAdvertisementRep (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

`virtual EcUtStatus FetchByValue (void)`

Privilege: Public
No Inheritance
This routine selects an object from the database based on its state. For contact, the unique application state is ProviderRole and OrganizationName.

`virtual EcUtStatus FetchPhaseII`
`(dataFromSelector:RWDBReader&)`

Privilege: Public
No Inheritance
This operation gets all the data from FetchPrep. This routine is part of the persistent framework.

`virtual void FetchPrep (dataToAcquire: RWDBSelector&, currentWhereClause: RWDBCriterion&, foreignKey: co nst RWDBExpr&)`

Privilege: Public
No Inheritance

`const char* GetAccessRestriction (void)`

Privilege: Public
No Inheritance
This operation returns a text description of access restrictions imposed by the provider.

`const char* GetOrganizationName (void)`

Privilege: Public
No Inheritance
This operation returns the organization name of the provider.

`const char* GetProviderRole (void)`

Privilege: Public
No Inheritance
This operation returns the name of the role of the provider.

`EcUtStatus InsertBasicData (void)`

Privilege: Protected Operation
No Inheritance
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

`void IoAdProviderRep (void)`

Privilege: Public
No Inheritance
Default constructor.

`void IoAdProviderRep (copyFrom: IoAdProviderRep&)`

Privilege: Public
No Inheritance
Copy constructor.

`virtual EctBoolean IsValid (void)`

Privilege: Public
No Inheritance
This operation checks the contents of the object for consistency and data rules: o Ad Base is valid   o Organization name and Provider role is not NULL

`virtual void PrintMembers (out: ostream&)`

Privilege: Public
No Inheritance
This operation writes contents to stream.

`const char* ProviderTableName (void)`

Privilege: Private
No Inheritance
This is a private function to encapsulate table name.

`void SetAccessRestriction (value: const char*)`

Privilege: Public
No Inheritance
This operation set a new text description of access restrictions imposed by the provider.

`void SetOrganizationName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the organization name of the provider.

`void SetProviderRole (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the role name of the provider.

```
 SetValues (title: const char*, description: const char*,
guideURL: const char*, group: cons t char*, ur: const char*,
copyRight: const char*, contact: IoAdContact, provide
rRole:const char*, organizationName: const char*,
accessRestriction: const char*)
```

Privilege: Protection Not Identified
No Inheritance

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID. This routine is part of the persistent framework.

```
const IoAdProviderRep& operator= (assignFrom:
IoAdProviderRep&)
```

Privilege: Public
No Inheritance
This operator enables the assignment of two objects.

```
void ~IoAdProviderRep (void)
```

Privilege: Public
No Inheritance
Default destructor.

5.1.2.10 Class IoAdProviderSearchCommand

Synopsis:

Parent Class: IoAdSearchCommand
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdSearchCommand

Description:

Public View: This class provides interfaces for applications to search the set of product advertisements by specifying options and criterion. The persistent data will be stored into a results list for additional searches or access. Users should set up options of how to search (filtering, patterns, how many results to return) and then call the search interfaces. Protected View: None. Private View: None.

Attributes:

`myResults`

Privilege: Private
Data Type: IoAdProviderList
Default Value: NOT IDENTIFIED
No Inheritance
This is the result that contains the matched Provider Advertisement.

`myAdvType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
What kind of search type (product, service, provider).

`myDescTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the description table of the database.

`myMasterTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the master table.

`myMatchType`

Privilege: Private
Data Type: MatchTypeEnum
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains types of pattern (Prefix/Contain/Exact) matching allowable on string searches.

`myPattern`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains the matched pattern to be searched.

`myResults`

Privilege: Private
Data Type: IoAdAdvertisementList
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains accummulated set of results for this search.

`mySearchLimit`

Privilege: Private
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
This is the match type (Prefix/Contain/Exact) that the pattern will be compared, or else all the matched will be found.

Operations:

`EcTVoid ClearResults (void)`

Privilege: Public
No Inheritance
This operation clears the matched object linked list.

`const IoAdProviderList& GetResults (void)`

Privilege: Public
No Inheritance
This operation returns the matched Provider Advertisement to user.

`void IoAdproviderSearchCommand (void)`

Privilege: Public
No Inheritance
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
No Inheritance
This operation resets all the search criterion to default. Default Adv type searches Provider Advertisement type.

`EcUtStatus SearchByOrgName (MatchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for Provider Advertisement that contains a substring of matchTo in its organization Name and appends found Product Advertisements to the current results set.

`EcTVoid SetSearchType (typeFilter:AdvertisingType)`

Privilege: Protected Operation
No Inheritance
This operation hides from users our base's SetSearchType method.

`void ~IoAdProviderSearchCommand (void)`

Privilege: Public
No Inheritance
Default destructor.

`EcTVoid ClearResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation clears the matched object linked list.

`RWDBCriterion CommonExpr (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Adv Type search.

`EcUtStatus CommonQueryProcessing (&select:RWDBSelectorBase)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation processes the search of persistent object list for pattern matched.

`RWDBSelector CommonSelector (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation constructs the search attribute for persistent objects.

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation returns the default database connection.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation connects to the database server.

`RWDBCriterion Expr (curTable:RWDBTable, curColumn:RWDBColumn)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion according to the matched pattern.

`RWDBCriterion ExprByDescription (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Description search.

`RWDBCriterion ExprByTitle (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Title search.

`RWDBCriterion ExprByURL (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for URL search.

`RWCString GetPattern (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation retrieves the matched pattern to be searched.

`const IoAdAdvertisementList& GetResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation returns the matched Advertisement object to user.

`EcUtStatus InsertAdvertisementIntoResults (id:EcTPDatabaseID, type: AdvertisingType)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation creates an advertisement with the characteristic specified.

`void IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation resets all the search criterion to default. Default Adv type searches all product/provider/service type, match Type is defaulted to prefix, and default search limit returns all.

`EcUtStatus SearchByText (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title, URL, or Description and appends found advertisements to the current results set.

`EcUtStatus SearchByTitle (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title and appends found advertisements to the current results set.

`EcTVoid SetMatchType (matchType:MatchTypeEnum)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared.

`EcTVoid SetPattern (matchTo:RWCString)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation stores the matched pattern to be searched.

`EcTVoid SetSearchLimit (maxToReturn:EcTUInt)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared or else all the matched will be found.

`EcTVoid SetSearchType (typeFilter:AdvertisingType)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets all the Adv Type desired, if not specified, all Adv Type will be returned.

`void ~IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default destructor.

5.1.2.11 Class IoAdSearchCommand

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

Public View: This class provides interfaces for applications to search the set of all derived classes of advertisements by specifying criterion. The persistent data will be stored into a results list for additional searches or access. Users should set up options of how to search (filtering, patterns, how many results to return) and then call the search interfaces. While concrete, derived objects are placed in the results list, we return a list of abstract ads objects. If one wants to access members of the derived type of ads, use the associated derived-ad-List class to filter the ad-list. For example, to see Product Ads, use the ProductList class. Protected View: None. Private View: None.

Attributes:

`myAdvType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
No Inheritance
What kind of search type (product, service, provider).

`myDescTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
No Inheritance
The name of the description table of the database.

`myMasterTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
No Inheritance
The name of the master table.

`myMatchType`

Privilege: Private
Data Type: MatchTypeEnum
Default Value: NOT IDENTIFIED
No Inheritance
Contains types of pattern (Prefix/Contain/Exact) matching allowable on string searches.

`myPattern`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Contains the matched pattern to be searched.

`myResults`

Privilege: Private
Data Type: IoAdAdvertisementList
Default Value: NOT IDENTIFIED
No Inheritance
Contains accummulated set of results for this search.

`mySearchLimit`

Privilege: Private
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
No Inheritance
This is the match type (Prefix/Contain/Exact) that the pattern will be compared, or else all the matched will be found.

Operations:

`EcTVoid ClearResults (void)`

Privilege: Public
No Inheritance
This operation clears the matched object linked list.

`RWDBCriterion CommonExpr (expr:RWDBCriterion)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for Adv Type search.

`EcUtStatus CommonQueryProcessing (&select:RWDBSelectorBase)`

Privilege: Private
No Inheritance
This operation processes the search of persistent object list for pattern matched.

`RWDBSelector CommonSelector (expr:RWDBCriterion)`

Privilege: Private
No Inheritance
This operation constructs the search attribute for persistent objects.

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
No Inheritance
This operation returns the default database connection.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
No Inheritance
This operation connects to the database server.

`RWDBCriterion Expr (curTable:RWDBTable, curColumn:RWDBColumn)`

Privilege: Private
No Inheritance
This operation formulates the search criterion according to the matched pattern.

`RWDBCriterion ExprByDescription (void)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for Description search.

`RWDBCriterion ExprByTitle (void)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for Title search.

`RWDBCriterion ExprByURL (void)`

Privilege: Private
No Inheritance
This operation formulates the search criterion for URL search.

`RWCString GetPattern (void)`

Privilege: Private
No Inheritance
This operation retrieves the matched pattern to be searched.

`const IoAdAdvertisementList& GetResults (void)`

Privilege: Public
No Inheritance
This operation returns the matched Advertisement object to user.

`EcUtStatus InsertAdvertisementIntoResults`
`(id:EcTPDatabaseID, type: AdvertisingType)`

Privilege: Private
No Inheritance
This operation creates an advertisement with the characteristic specified.

`void IoAdSearchCommand (void)`

Privilege: Public
No Inheritance
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
No Inheritance
This operation resets all the search criterion to default. Default Adv type searches all product/provider/service type, match Type is defaulted to prefix, and default search limit returns all.

`EcUtStatus SearchByText (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for any Advertisement that contains a substring of matchTo in its Title, URL, or Description and appends found advertisements to the current results set.

`EcUtStatus SearchByTitle (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for any Advertisement that contains a substring of matchTo in its Title and appends found advertisements to the current results set.

`EcTVoid SetMatchType (matchType:MatchTypeEnum)`

Privilege: Public
No Inheritance
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared.

`EcTVoid SetPattern (matchTo:RWCString)`

Privilege: Private
No Inheritance
This operation stores the matched pattern to be searched.

`EcTVoid SetSearchLimit (maxToReturn:EcTUInt)`

Privilege: Public
No Inheritance
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared or else all the matched will be found.

`EcTVoid SetSearchType (typeFilter:AdvertisingType)`

Privilege: Public
No Inheritance
This operation sets all the Adv Type desired, if not specified, all Adv Type will be returned.

`void ~IoAdSearchCommand (void)`

Privilege: Public
No Inheritance
Default destructor.

5.1.2.12 Class IoAdService

Synopsis:

Parent Class: IoAdAdvertisement
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdAdvertisement

Description:

Public View: This entity class supports operations to allow the definition, storage and retrieval of a advertisement of a service. Typically, this service processes ECS data products. It can also be other kinds of automated services or non automated services (e.g. help desk). Member data and operations should not be accessed directly, but through IoAdService. IoAdService performs handle services. Because fetching is deferred, any access can generate a database error. Therefore, one should check the handles status when appropriate. Unless specified, the string values can be empty. Protected View: We inherit our Database() and Connection() from IoAdAdvertisement. Private View: None.

Attributes:

`myContact`

Privilege: Private
Data Type: IoAdContact
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Who/What is responsible for the advertised entity.

`myCopyRight`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Any relevant copyright that applies to the entity being advertised.

`myDescription`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Long textual description of the advertised entity.

`myExpirationDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When am I no longer valid.

`myGroup`

Privilege: Private
Data Type: RWCString myGroup
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Logical group I am part of for administration.

`myGuideURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
The Web URL for a guide to my entity.

`myStartDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When I am valid for advertisement.

`myTitle`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Stores the unique description for me.

`myType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
What derived type are we part of.

`myUR`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
A universal reference to access my advertised entity.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

`myProducts`

Privilege: Private
Data Type: IoAdProductReferenceList
Default Value: NOT IDENTIFIED
No Inheritance
What products can we apply to.

`myProvider`

Privilege: Private
Data Type: IoAdProvider
Default Value: NOT IDENTIFIED
No Inheritance
Who is providing this product.

`myServiceClass`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the name of the service class for the advertised service.

`myServiceTypeId`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Stores the signature of the service.

`myServideName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Stores the name of the advertised service.

Operations:

`virtual RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid connection.

`virtual RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid database with data in it.

`static const char* DescriptionTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

```
virtual EcUtStatus FetchPhaseII
(dataFromSelector:RWDBReader&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on, foreignKey:const RWDBExpr&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation prepares the selector to acquire all of classes data.

`IoAdContact GetContact (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets who/what is responsible for the advertised entity.

```
const char* GetCopyRight (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets any relevant copyrights that apply to the entity being advertised.

```
const char* GetDescription (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a long textual description of the advertised entity.

```
const char* GetGroup (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the logical group I am part of for administration.

```
const char* GetGuideURL (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the Web URL for a guide to my entity.

```
RWDBDateTime GetStartDate (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the start date when the ad is valid.

```
const char* GetTitle (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a unique description of the title .

```
AdvertisingType GetType (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the derived type that the advertisement is part of.

```
const char* GetUR (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the universal reference to access the advertised entity.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation inserts our data if it is logically valid.

`void IoAdAdvertisementRep (type:AdvertisingType)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Default constructor.

`void IoAdAdvertisementRep (copyFrom:IoAdAdvertisementRep&)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Copy constructor.

`EcTBoolean IsValid (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation checks the contents of the object for consistency and data rules: o Contact is valid o Title, Description, and Group are not NULL

`static const char* MasterTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation writes contents to stream.

`void SetContact (value:IoAdContact&)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets who/what is responsible for the advertised entity.

```
void SetCopyRight (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets any relevant copyrights that apply to the entity being advertised.

```
void SetDescription (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the long textual description of the advertised entity.

```
void SetGroup (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the logical group I am part of for administration.

```
void SetGuideURL (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the Web URL for a guide to my entity.

```
void SetStartDate (value:RWDBDateTime)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the start date when the ad is valid.

```
void SetTitle (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets a unique description of the title.

```
void SetUR (value:const char)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the universal reference to access the advertised entity.

```
void SetValues (type:AdvertisingType,title:const
char*,description:const char*, guideURL:const
char*,group:const char*,UR:const char*,copyRight:const
char*, contact:IoAdContact&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation sets values for the parameters passed in the argument lists.

```
static const char* URTableName (void)
```

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contains names to encapsulate database table.

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation updates this object in the database for its current object ID.

```
IoAdAdvertisementRep* operator-> (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation enables the -> operation to access members of the base class IoAdAdvertisementRep.

```
const IoAdAdvertisementRep& operator=
(assignFrom:IoAdAdvertisementRep&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operator enables the assignment of two objects.

```
void ~IoAdAdvertisement (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

```
void ~IoAdAdvertisementRep (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

`EcUtStatus AddProduct (productToAdd:IoAdProduct)`

Privilege: Public
No Inheritance
This operation defines a new product applies to this service. This in turn also defines this service is applicable to the given product.

`EcUtStatus DeleteProduct (productToDelete:IoAdProduct)`

Privilege: Public
No Inheritance
This operation removes a defined product from this service. This in turn also removes this service from the product.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
No Inheritance
This routine selects an object from the database based on its state. For contact, the unique applicable state is ServiceClassID, ServiceName, and ServiceID

```
virtual EcUtStatus FetchPhaseII
(dataFromSelector:RWDBReader&)
```

Privilege: Public
No Inheritance
This operation gets all the data from FetchPrep.

```
 FetchPrep (dataToAcquire:RWDBSelector&,
currentWhereClause:RWDBCriterion&, foreignKey:const
RWDBExpr&)
```

Privilege: Protection Not Identified
No Inheritance
This operation prepares the selector to acquire all of classes data.

`IoAdProvider GetProvider (void)`

Privilege: Public
No Inheritance
This operation gets the provider who is providing the product.

`const char* GetServiceClass (void)`

Privilege: Public
No Inheritance
This operation returns the current service class name of the advertised service.

`const char* GetServiceName (void)`

Privilege: Public
No Inheritance
This operation returns the name of the advertised service.

`EcTInt GetServiceTypeId (void)`

Privilege: Public
No Inheritance
This operation defines the signature of the service.

`EcUtStatus InsertBasicData (void)`

Privilege: Protected Operation
No Inheritance
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

`void IoAdServiceRep (void)`

Privilege: Public
No Inheritance
This constructor sets our advertising type. It initializes our list of products to know we are the source object.

`void IoAdServiceRep (copyFrom:IoAdServiceRep&)`

Privilege: Public
No Inheritance
Copy constructor.

`virtual EcTBoolean IsValid (void)`

Privilege: Public
No Inheritance
This operation checks the contents of the object for consistency and data rules.   o Provider is valid   o Ad Base is valid   o Service class and service name are not NULL

`EcTUInt NumberOfProducts (void)`

Privilege: Public
No Inheritance
This operation defines the number of products that service applies to.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
No Inheritance
This operation writes contents to stream.

`IoAdProductReferenceListIter ProductIter (void)`

Privilege: Public
No Inheritance
This operation provides a RWTPSlist-like iterator for all applicable products.

`const char* ServiceTableName (void)`

Privilege: Private
No Inheritance
This is a private function to encapsulate database table.

`void SetProvider (value:IoAdProvider)`

Privilege: Public
No Inheritance
This operation sets the myProvider attribute.

`void SetServiceClass (value:const char*)`

Privilege: Public
No Inheritance
This operation is used to set a new service class name for the advertised service.

`void SetServiceName (value:const char*)`

Privilege: Public
No Inheritance
This operation is used to set the name of the advertised service.

`void SetServiceTypeId (value:EcTInt)`

Privilege: Public
No Inheritance
This operation sets the signature of the service.

```
 SetValues (title: const char*, description: const char*,
guideURL const char*, group: cons t char*, ur:const char*,
copyRight:const char*, contact: IoAdContact, provider:
IoAdProvider, serviceClass: const char*, serviceName:const
char*, serviceType Id: EcTInt)
```

Privilege: Protection Not Identified
No Inheritance

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID. This routine is part of the persistent framework.

```
const IoAdServiceRep& operator= (assignFrom:IoAdServiceRep&)
```

Privilege: Public
No Inheritance
This operator enables the assignment of two objects.

```
void ~IoAdServiceRep (void)
```

Privilege: Public
No Inheritance
Default destructor.

5.1.2.13 Class IoAdServiceSearchCommand

Synopsis:

Parent Class: IoAdSearchCommand
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdSearchCommand

Description:

Public View: This class provides interfaces for applications to search the set of service advertisements by specifying options and criterion. The persistent data will be stored into a results list for additional searches or access. Users should set up options of how to search (filtering, patterns, how many results to return) and then call the search interfaces. Protected View: None. Private View: None.

Attributes:

```
myResults
```

Privilege: Private
Data Type: IoAdServiceList
Default Value: NOT IDENTIFIED
No Inheritance
This is the result that contains the matched Service Advertisement.

`myAdvType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
What kind of search type (product, service, provider).

`myDescTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the description table of the database.

`myMasterTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the master table.

`myMatchType`

Privilege: Private
Data Type: MatchTypeEnum
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains types of pattern (Prefix/Contain/Exact) matching allowable on string searches.

`myPattern`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains the matched pattern to be searched.

`myResults`

Privilege: Private
Data Type: IoAdAdvertisementList
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains accummulated set of results for this search.

`mySearchLimit`

Privilege: Private
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
This is the match type (Prefix/Contain/Exact) that the pattern will be compared, or else all the matched will be found.

Operations:

`EcTVoid ClearResults (void)`

Privilege: Public
No Inheritance
This operation clears the matched object linked list.

`const IoAdServiceList& GetResults (void)`

Privilege: Public
No Inheritance
This operation returns the matched Service Advertisement to user.

`void IoAdServiceSearchCommand (void)`

Privilege: Public
No Inheritance
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
No Inheritance
This operation resets all the search criterion to default. Default Adv type searches Service Advertisement type.

`EcUtStatus SearchByServiceName (matchTo:const char*)`

Privilege: Public
No Inheritance
This operation searches for Service Advertisement that contains a substring of matchTo in its ServiceName and appends found Service advertisements to the current results set.

`EcTVoid SetSearchType (typeFilter:AdvertisingType)`

Privilege: Protected Operation
No Inheritance
This operation sets the type of search .

`void ~IoAdServiceSearchCommand (void)`

Privilege: Public
No Inheritance
Default destructor.

`EcTVoid ClearResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation clears the matched object linked list.

`RWDBCriterion CommonExpr (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Adv Type search.

`EcUtStatus CommonQueryProcessing (&select:RWDBSelectorBase)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation processes the search of persistent object list for pattern matched.

`RWDBSelector CommonSelector (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation constructs the search attribute for persistent objects.

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation returns the default database connection.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation connects to the database server.

```
RWDBCriterion Expr (curTable:RWDBTable,
curColumn:RWDBColumn)
```

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion according to the matched pattern.

`RWDBCriterion ExprByDescription (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Description search.

`RWDBCriterion ExprByTitle (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Title search.

`RWDBCriterion ExprByURL (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for URL search.

`RWCString GetPattern (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation retrieves the matched pattern to be searched.

`const IoAdAdvertisementList& GetResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation returns the matched Advertisement object to user.

```
EcUtStatus InsertAdvertisementIntoResults
(id:EcTPDatabaseID, type: AdvertisingType)
```

Privilege: Private
Inherited From: IoAdSearchCommand
This operation creates an advertisement with the characteristic specified.

`void IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation resets all the search criterion to default. Default Adv type searches all product/provider/service type, match Type is defaulted to prefix, and default search limit returns all.

`EcUtStatus SearchByText (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title, URL, or Description and appends found advertisements to the current results set.

`EcUtStatus SearchByTitle (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title and appends found advertisements to the current results set.

`EcTVoid SetMatchType (matchType:MatchTypeEnum)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared.

`EcTVoid SetPattern (matchTo:RWCString)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation stores the matched pattern to be searched.

`EcTVoid SetSearchLimit (maxToReturn:EcTUInt)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared or else all the matched will be found.

`EcTVoid SetSearchType (typeFilter:AdvertisingType)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets all the Adv Type desired, if not specified, all Adv Type will be returned.

`void ~IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default destructor.

5.1.2.14 Class IoAdSignatureServiceAdv

Synopsis:

Parent Class: IoAdService
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdService

Description:

Public View: This entity class supports operations to allow the definition, storage and retrieval of an advertisement of a signature service. Member data and operations should not be accessed directly, but through IoAdSignatureServiceAdv. IoAdSignatureServiceAdv performs handle services. Because fetching is deferred, any access can gernerate a database error. Therefore, one should check the handles status when appropriate. Unless specified, the string values can be empty. Protected View: We inherit our Database() and Connection() from IoAdAdvertisement. Private View: None

Attributes:

`mySignatureServiceSchema`

Privilege: Private
Data Type: IoAdSignatureServiceSchema
Default Value: NOT IDENTIFIED
No Inheritance
Which Signature Service Schema is associated with.

`myContact`

Privilege: Private
Data Type: IoAdContact
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Who/What is responsible for the advertised entity.

`myCopyRight`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Any relevant copyright that applies to the entity being advertised.

`myDescription`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Long textual description of the advertised entity.

`myExpirationDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When am I no longer valid.

`myGroup`

Privilege: Private
Data Type: RWCString myGroup
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Logical group I am part of for administration.

`myGuideURL`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
The Web URL for a guide to my entity.

`myStartDate`

Privilege: Private
Data Type: RWDBDateTime
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
When I am valid for advertisement.

`myTitle`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
Stores the unique description for me.

`myType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
What derived type are we part of.

`myUR`

Privilege: Private
Data Type: EcPoLongString
Default Value: NOT IDENTIFIED
Inherited From: IoAdAdvertisement
A universal reference to access my advertised entity.

`myRep`

Privilege: Protected Attribute
Data Type: EcPoPersistentBase*
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current concrete representation.

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcPoHandle
This is the current status.

`myProducts`

Privilege: Private
Data Type: IoAdProductReferenceList
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
What products can we apply to.

`myProvider`

Privilege: Private
Data Type: IoAdProvider
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Who is providing this product.

`myServiceClass`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Stores the name of the service class for the advertised service.

`myServiceTypeId`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Stores the signature of the service.

`myServideName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdService
Stores the name of the advertised service.

Operations:

`EcUtStatus DeleteBasicData (void)`

Privilege: Private
No Inheritance
This operation deletes all attributes in database.

`virtual EcUtStatus DeleteDerived (void)`

Privilege: Protected Operation
No Inheritance
This operation deletes this object in the database for its current object ID. This routine is part of the persistent framework.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
No Inheritance
This routine selects an object from the database based on its state. For contact, the unique application state is ServiceClassID, ServiceName, and ServiceID.

`virtual EcUtStatus FetchPhaseII (dataFromSelector : RWDBReader&)`

Privilege: Public
No Inheritance
This operation gets all the data from FetchPrep.

`FetchPrep (dataToAcquire: RWDBSelector& , currentWhereClause : RWDBCriterion, foreignKey : const RWDBExpr&)`

Privilege: Protection Not Identified
No Inheritance

`const char* GetServiceClass (void)`

Privilege: Public
No Inheritance
This operation gets the service class type supplied by the subsystem.

`const char* GetServiceName (void)`

Privilege: Public
No Inheritance
This operator gets the service operator for particular service class type such as Ingest/Ceres02 or Delete/Ceres02.

`const IoAdSignatureServiceSchema* GetSignatureServiceSchema (void)`

Privilege: Public
No Inheritance
This operation gets the signature service schema .

`virtual EcUtStatus Insert Derived (void)`

Privilege: Protected Operation
No Inheritance
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
No Inheritance
This operation stores all attributes in database.

`void IoAdSignatureServiceAdvRep (void)`

Privilege: Public
No Inheritance
This constructor is responsible for initializing the IoAdSignatureServiceAdvRep class.

`void IoAdSignatureServiceAdvRep (copyFrom : ioAdSignatureServiceAdvRep&)`

Privilege: Public
No Inheritance
Copy constructor.

`virtual EcTBoolean IsValid (void)`

Privilege: Public
No Inheritance
This operation checks the contents of the object for consistency and data rules: o Provider is valid o Ad Base is valid o Service class and service name are not NULL

`virtual void PrintMembers (out: ostream&)`

Privilege: Public
No Inheritance
This operation writes contents to stream.

`void SetServiceClass (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the service class type supplied by the subsystem.

`void SetServiceName (value: const char*)`

Privilege: Public
No Inheritance
This operation sets the service operator for particular service class type such as Ingest/Ceres02 or Delete/Ceres02.

```
void SetSignatureServiceSchema (value:const
IoAdSignatureServiceSchema)
```

Privilege: Public
No Inheritance
This operation sets the signature service schema.

```
 SetValues (title: const char*, description: const char*,
guideURL: const char*, group : const char*, ur: const char*,
copyRight: const char*, contact : IoAdContact, provider:
IoAdProvider, serviceClass: const char*, serviceName: const
char*, serviceSchema: IoAdSignatureServiceSchema)
```

Privilege: Protection Not Identified
No Inheritance

```
void SignatureServiceAdvTable (void)
```

Privilege: Public
No Inheritance
This operation stores the signature service table name stored in the database.

```
EcUtStatus UpdateBasicData (void)
```

Privilege: Private
No Inheritance
This operation stores all attributes in database.

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
No Inheritance
This operation updates this object in the database for its current object ID. This routine is part of the persistent framework.

```
const IoAdSignatureServiceAdvRep& operator= (assignFrom :
IoAdSignatureServiceAdvRep&)
```

Privilege: Public
No Inheritance
This operation enables the assignment of two objects.

```
void ~IoAdSignatureServiceAdvRep (void)
```

Privilege: Public
No Inheritance
Default destructor.

```
virtual RWDBConnection& Connection (void)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid connection.

```
virtual RWDBDatabase& Database (void)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation returns a valid database with data in it.

```
static const char* DescriptionTableName (void)
```

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

```
virtual EcUtStatus FetchPhaseII
(dataFromSelector:RWDBReader&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation gets all the data from FetchPrep.

```
virtual void FetchPrep
(dataToAcquire:RWDBSelector&,currentWhereClause:RWDBCriteri
on, foreignKey:const RWDBExpr&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation prepares the selector to acquire all of classes data.

```
IoAdContact GetContact (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets who/what is responsible for the advertised entity.

```
const char* GetCopyRight (void)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets any relevant copyrights that apply to the entity being advertised.

`const char* GetDescription (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a long textual description of the advertised entity.

`const char* GetGroup (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the logical group I am part of for administration.

`const char* GetGuideURL (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the Web URL for a guide to my entity.

`RWDBDateTime GetStartDate (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the start date when the ad is valid.

`const char* GetTitle (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets a unique description of the title .

`AdvertisingType GetType (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the derived type that the advertisement is part of.

`const char* GetUR (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation gets the universal reference to access the advertised entity.

`EcUtStatus InsertBasicData (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation stores all attributes in database.

```
virtual EcUtStatus InsertDerived (void)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation inserts our data if it is logically valid.

```
void IoAdAdvertisementRep (type:AdvertisingType)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Default constructor.

```
void IoAdAdvertisementRep (copyFrom:IoAdAdvertisementRep&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
Copy constructor.

```
EcTBoolean IsValid (void)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation checks the contents of the object for consistency and data rules: o Contact is valid o Title, Description, and Group are not NULL

```
static const char* MasterTableName (void)
```

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contain names to encapsulate database table.

```
virtual void PrintMembers (out:ostream&)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation writes contents to stream.

```
void SetContact (value:IoAdContact&)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets who/what is responsible for the advertised entity.

```
void SetCopyRight (value:const char*)
```

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets any relevant copyrights that apply to the entity being advertised.

`void SetDescription (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the long textual description of the advertised entity.

`void SetGroup (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the logical group I am part of for administration.

`void SetGuideURL (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the Web URL for a guide to my entity.

`void SetStartDate (value:RWDBDateTime)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the start date when the ad is valid.

`void SetTitle (value:const char*)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets a unique description of the title.

`void SetUR (value:const char)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation sets the universal reference to access the advertised entity.

```
void SetValues (type:AdvertisingType,title:const
char*,description:const char*, guideURL:const
char*,group:const char*,UR:const char*,copyRight:const
char*, contact:IoAdContact&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation sets values for the parameters passed in the argument lists.

`static const char* URTableName (void)`

Privilege: Private
Inherited From: IoAdAdvertisement
This operation contains names to encapsulate database table.

`virtual EcUtStatus UpdateDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operation updates this object in the database for its current object ID.

`IoAdAdvertisementRep* operator-> (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
This operation enables the -> operation to access members of the base class IoAdAdvertisementRep.

```
const IoAdAdvertisementRep& operator=
(assignFrom:IoAdAdvertisementRep&)
```

Privilege: Protected Operation
Inherited From: IoAdAdvertisement
This operator enables the assignment of two objects.

`void ~IoAdAdvertisement (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void ~IoAdAdvertisementRep (void)`

Privilege: Public
Inherited From: IoAdAdvertisement
Default destructor.

`void Clone (cloneFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation makes this handle use a new copy (clone) of an existing representation. Since copy does representation sharing, this is the only way to get a logical duplicate of the object.

`void EcPoHandle (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Default constructor. It creates this object with a new representation.

`void EcPoHandle (copyFrom:EcPoHandle&)`

Privilege: Protected Operation
Inherited From: EcPoHandle
Copy constructor. It creates this object by sharing the representation of another handle.

`void Fetch (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation binds this object to a database object specified by IDToMatch and "logically" load the data from that object into this object.

`void FetchNow (IDToMatch: const EcTPoDatabaseID&, readonly = EcDFalse: EcTBoolean)`

Privilege: Public
Inherited From: EcPoHandle
This operation is the same as matching Fetch, but WILL NOT defer. It gets the data for this object, based on the argument ID.

`RWTPtrSlist<EcPoPersistentBase>& FetchedObjects (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns the container of concrete cached objects.

`void FinishFetch (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation forces the completion of any pending fetches. Derived handle class referencing representation data should always call this first.

`EcUtStatus GetStatus (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation returns the status for the object's data. It checks if the object's data is valid or not. If the status is OK, it forces a database fetch.

`EcPoPersistentBase* NewRep (void)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation.

`EcPoPersistentBase* NewRep (copyFrom: const EcPoPersistentBase*)`

Privilege: Protected Operation
Inherited From: EcPoHandle
This operation returns a new concrete representation based on an existing one.

`void Store (void)`

Privilege: Public
Inherited From: EcPoHandle
This operation stores the current data to the database. If this object has never been in the database, does an insert, otherwise, does an update. The status object will contain our validity information.

`EcPoPersistentBase* operator-> (void)`

Privilege: Public
Inherited From: EcPoHandle
If the client is looking for a PersistentBase representation, this operation returns it.

`const EcPoHandle& operator= (assignFrom: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation resets this object handle to share another handle's representation.

`EcTBoolean operator== (equalTo: EcPoHandle&)`

Privilege: Public
Inherited From: EcPoHandle
This operation compares the representations of this and another handle.

`EcUtStatus AddProduct (productToAdd:IoAdProduct)`

Privilege: Public
Inherited From: IoAdService
This operation defines a new product applies to this service. This in turn also defines this service is applicable to the given product.

`EcUtStatus DeleteProduct (productToDelete:IoAdProduct)`

Privilege: Public
Inherited From: IoAdService
This operation removes a defined product from this service. This in turn also removes this service from the product.

`virtual EcUtStatus FetchByValues (void)`

Privilege: Public
Inherited From: IoAdService
This routine selects an object from the database based on its state. For contact, the unique applicable state is ServiceClassID, ServiceName, and ServiceID

`virtual EcUtStatus FetchPhaseII (dataFromSelector:RWDBReader&)`

Privilege: Public
Inherited From: IoAdService
This operation gets all the data from FetchPrep.

`FetchPrep (dataToAcquire:RWDBSelector&, currentWhereClause:RWDBCriterion&, foreignKey:const RWDBExpr&)`

Privilege: Protection Not Identified
Inherited From: IoAdService
This operation prepares the selector to acquire all of classes data.

`IoAdProvider GetProvider (void)`

Privilege: Public
Inherited From: IoAdService
This operation gets the provider who is providing the product.

`const char* GetServiceClass (void)`

Privilege: Public
Inherited From: IoAdService
This operation returns the current service class name of the advertised service.

`const char* GetServiceName (void)`

Privilege: Public
Inherited From: IoAdService
This operation returns the name of the advertised service.

`EcTInt GetServiceTypeId (void)`

Privilege: Public
Inherited From: IoAdService
This operation defines the signature of the service.

`EcUtStatus InsertBasicData (void)`

Privilege: Protected Operation
Inherited From: IoAdService
This operation stores all attributes in database.

`virtual EcUtStatus InsertDerived (void)`

Privilege: Protected Operation
Inherited From: IoAdService
This operation inserts our data if it is logically valid. This routine is part of the persistent framework.

`void IoAdServiceRep (void)`

Privilege: Public
Inherited From: IoAdService
This constructor sets our advertising type. It initializes our list of products to know we are the source object.

`void IoAdServiceRep (copyFrom:IoAdServiceRep&)`

Privilege: Public
Inherited From: IoAdService
Copy constructor.

`virtual EcTBoolean IsValid (void)`

Privilege: Public
Inherited From: IoAdService
This operation checks the contents of the object for consistency and data rules. o Provider is valid o Ad Base is valid o Service class and service name are not NULL

`EcTUInt NumberOfProducts (void)`

Privilege: Public
Inherited From: IoAdService
This operation defines the number of products that service applies to.

`virtual void PrintMembers (out:ostream&)`

Privilege: Public
Inherited From: IoAdService
This operation writes contents to stream.

`IoAdProductReferenceListIter ProductIter (void)`

Privilege: Public
Inherited From: IoAdService
This operation provides a RWTPSlist-like iterator for all applicable products.

`const char* ServiceTableName (void)`

Privilege: Private
Inherited From: IoAdService
This is a private function to encapsulate database table.

`void SetProvider (value:IoAdProvider)`

Privilege: Public
Inherited From: IoAdService
This operation sets the myProvider attribute.

`void SetServiceClass (value:const char*)`

Privilege: Public
Inherited From: IoAdService
This operation is used to set a new service class name for the advertised service.

`void SetServiceName (value:const char*)`

Privilege: Public
Inherited From: IoAdService
This operation is used to set the name of the advertised service.

`void SetServiceTypeId (value:EcTInt)`

Privilege: Public
Inherited From: IoAdService
This operation sets the signature of the service.

```
SetValues (title: const char*, description: const char*,
guideURL const char*, group: cons t char*, ur:const char*,
copyRight:const char*, contact: IoAdContact, provider:
IoAdProvider, serviceClass: const char*, serviceName:const
char*, serviceType Id: EcTInt)
```

Privilege: Protection Not Identified
Inherited From: IoAdService

```
virtual EcUtStatus UpdateDerived (void)
```

Privilege: Protected Operation
Inherited From: IoAdService
This operation updates this object in the database for its current object ID. This routine is part of the persistent framework.

```
const IoAdServiceRep& operator= (assignFrom:IoAdServiceRep&)
```

Privilege: Public
Inherited From: IoAdService
This operator enables the assignment of two objects.

```
void ~IoAdServiceRep (void)
```

Privilege: Public
Inherited From: IoAdService
Default destructor.

5.1.2.15 Class IoAdSignatureServiceSearchCommand

Synopsis:

Parent Class: IoAdSearchCommand
Is Not A Distributed Object
Is Associated With:
This class is derived from the class IoAdSearchCommand

Description:

Public View: This class provides interfaces for applications to search the set of Signature type service advertisements by specifying options and criterion. The persistent data will be stored into a results list for additional searches or access. Users should set up options of how to search (filtering, patterns, how many results to return) and then call the search interfaces. Protected View: None. Private View: None.

Attributes:

`myResults`

Privilege: Private
Data Type: IoAdSignatureServiceList
Default Value: NOT IDENTIFIED
No Inheritance
This is the found Signature type Service lists.

`myAdvType`

Privilege: Private
Data Type: AdvertisingType
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
What kind of search type (product, service, provider).

`myDescTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the description table of the database.

`myMasterTable`

Privilege: Private
Data Type: RWDBTable
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
The name of the master table.

`myMatchType`

Privilege: Private
Data Type: MatchTypeEnum
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains types of pattern (Prefix/Contain/Exact) matching allowable on string searches.

`myPattern`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains the matched pattern to be searched.

`myResults`

Privilege: Private
Data Type: IoAdAdvertisementList
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
Contains accummulated set of results for this search.

`mySearchLimit`

Privilege: Private
Data Type: EcTUInt
Default Value: NOT IDENTIFIED
Inherited From: IoAdSearchCommand
This is the match type (Prefix/Contain/Exact) that the pattern will be compared, or else all the matched will be found.

Operations:

`EcTVoid ClearResults (void)`

Privilege: Public
No Inheritance
This operation clears the matched object linked list.

`const IoAdSignatureServiceList & GetResults (void)`

Privilege: Public
No Inheritance
This operation returns the matched Service Advertisement to user.

`void IoAdSignatureServiceSearchCommand (void)`

Privilege: Public
No Inheritance
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
No Inheritance
This operation resets all the search options to default.   o Match type is "Prefix"   o No search limit on rows to match.

`EcUtStatus SearchByServiceClass (matchTo: const char*)`

Privilege: Public
No Inheritance
This operation searches for Service Advertisement that contains a substring of matchTo in its ServiceClass and append found Service advertisements to the current results set.

`EcUtStatus SearchByServiceName (matchTo: const char*)`

Privilege: Public
No Inheritance
This operation searches for Service Advertisement that contains a substring of matchTo in its ServiceName and append found Service advertisements to the current results set.

`void ~IoAdSignatureServiceSearchCommand (void)`

Privilege: Public
No Inheritance
Default destructor.

`EcTVoid ClearResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation clears the matched object linked list.

`RWDBCriterion CommonExpr (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Adv Type search.

`EcUtStatus CommonQueryProcessing (&select:RWDBSelectorBase)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation processes the search of persistent object list for pattern matched.

`RWDBSelector CommonSelector (expr:RWDBCriterion)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation constructs the search attribute for persistent objects.

`RWDBConnection& Connection (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation returns the default database connection.

`RWDBDatabase& Database (void)`

Privilege: Protected Operation
Inherited From: IoAdSearchCommand
This operation connects to the database server.

`RWDBCriterion Expr (curTable:RWDBTable, curColumn:RWDBColumn)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion according to the matched pattern.

`RWDBCriterion ExprByDescription (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Description search.

`RWDBCriterion ExprByTitle (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for Title search.

`RWDBCriterion ExprByURL (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation formulates the search criterion for URL search.

`RWCString GetPattern (void)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation retrieves the matched pattern to be searched.

`const IoAdAdvertisementList& GetResults (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation returns the matched Advertisement object to user.

`EcUtStatus InsertAdvertisementIntoResults (id:EcTPDatabaseID, type: AdvertisingType)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation creates an advertisement with the characteristic specified.

`void IoAdSearchCommand (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
Default constructor.

`EcTVoid Reset (void)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation resets all the search criterion to default. Default Adv type searches all product/provider/service type, match Type is defaulted to prefix, and default search limit returns all.

`EcUtStatus SearchByText (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title, URL, or Description and appends found advertisements to the current results set.

`EcUtStatus SearchByTitle (matchTo:const char*)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation searches for any Advertisement that contains a substring of matchTo in its Title and appends found advertisements to the current results set.

`EcTVoid SetMatchType (matchType:MatchTypeEnum)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared.

`EcTVoid SetPattern (matchTo:RWCString)`

Privilege: Private
Inherited From: IoAdSearchCommand
This operation stores the matched pattern to be searched.

`EcTVoid SetSearchLimit (maxToReturn:EcTUInt)`

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets the match type (Prefix/Contain/Exact) that the pattern will be compared or else all the matched will be found.

```
EcTVoid SetSearchType (typeFilter:AdvertisingType)
```

Privilege: Public
Inherited From: IoAdSearchCommand
This operation sets all the Adv Type desired, if not specified, all Adv Type will be returned.

```
void ~IoAdSearchCommand (void)
```

Privilege: Public
Inherited From: IoAdSearchCommand
Default destructor.

5.2 Communication Subsystem Classes

5.2.1 Communication Subsystem Classes Overview

CSS public classes constitute a basket of infrastructural services, characterized as "middleware," for use in all the subsystems (MSS, DSS, DPS, FOS, Ingest) of ECS. This infrastructure consists of communication services that application developers will use in the development of distributed applications, which enables these applications to interact with other applications within and outside of ECS. These services are standards-based and are interoperable (hardware and vendor independent). These services are broadly classified into three categories:

1. Common Facilities
2. Object Services
3. Distributed Object Framework (DOF)

The public classes in Common Facilities provide applications with legacy communications services required within the ECS infrastructure for file transfer (FTP), electronic mail, and bulletin board. In addition, a set of public classes which constitute generic event logger API are also included so that both application events as well as system management events can be logged into log files for subsequent analysis. The classes in Object Services support all ECS applications with interprocess communication and specialized infrastructural services such as security, directory, time, and asynchronous message passing. The Distributed Object Framework is an encapsulation of core object services, collectively providing object-oriented client server development and interaction amongst applications.

P ftp and DCE core services: Directory, Security, Time, RPCs. Release A provides mail, bulletin board, event logger, Message Passing and object oriented DCE services along with some enhancements.

Release B will use multiple cells with additional functionality: Secured Web and Transaction processing.

Addiitionl services provided by Release A include the following:

common facilities, mail, bulletin board and event log

5.2.2 Communication Subsystem Class Descriptions

5.2.2.1 Class CsBBMailRelA

Synopsis:

Parent Class: CsEmMailRelA
Is Not A Distributed Object
Is Associated With:
This class is derived from the class CsEmMailRelA

Description:

Used to post messages to bulletin boards.

Attributes:

`NNTPHost`

Privilege: Private
Data Type: EcTChar[EcDShortStr]
Default Value: NOT IDENTIFIED
No Inheritance
Will hold the name of the NNTP host to send the message to.

`myAttachedFiles`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
Inherited From: CsEmMailRelA
Internal stored (possibly uuencoded) message bodies.

`myBCCList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
Inherited From: CsEmMailRelA
Internal list of BCC recipients of the message.

`myCCList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
Inherited From: CsEmMailRelA
Internal list of CC's recipients of the message.

`myHeaderList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
Inherited From: CsEmMailRelA
Internal list of headers.

`myMessageBody`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
Inherited From: CsEmMailRelA
Internal copy of the message body.

`myToList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
Inherited From: CsEmMailRelA
Internal list of recipients of the message.

Operations:

`void CsBBMailRelA ()`

Privilege: Public
No Inheritance
Constructor that will create object and send the specified message. It will not delete any storage after the message has been sent.

`EcUtStatus Send ()`

Privilege: Public
No Inheritance
Will send the message in the current state. It will not delete any of the internally stored data.

`EcUtStatus SetNNTPHost (str:EcTChar*)`

Privilege: Public
No Inheritance
Will set the NNTP host to post the message to.

`void ~CsBBMailRelA ()`

Privilege: Public
No Inheritance
Destructor.

`EcUtStatus AddBCC (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will add the name string to the list of BCC'd recipients. Commas may be used to separate multiple usernames.

`EcUtStatus AddCC (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will add the name string to the list of CC'd recipients. Commas may be used to separate multiple usernames.

`EcUtStatus AddHeader (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will add an arbitrary header line to the message, the entire line you wish to appear in the header must be specified. Multiple calls will add multiple headers.

`EcUtStatus AddMessage (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will add the text to the currently stored body of the message. If there are attached files that have been already attached, calls to the AddMessage will add text after the file.

`EcUtStatus AddTo (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will add the name string to the list of recipients on the 'To:' line. Commas may be used to separate multiple usernames.

`EcUtStatus AttachFile (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will add the file named to the internal file storage. The file will be read in at the point of this call. The file may be uuencoded for transmittal.

`void CsEmMailRelA ()`

Privilege: Public
Inherited From: CsEmMailRelA
This is the default constructor. It will not fill in any of the user-settable fields.

`EcUtStatus Send ()`

Privilege: Public
Inherited From: CsEmMailRelA
Will send the message in the current state. It will not delete any of the internally stored data.

`EcUtStatus Subject (str:EcTChar*)`

Privilege: Public
Inherited From: CsEmMailRelA
Will set the subject to the passed text string. Will delete any previously created subjects.

`void ~CsEmMailRelA ()`

Privilege: Public
Inherited From: CsEmMailRelA
Will delete the object and all internal storage used. If mailing large files, it is recommended that this be performed as soon as possible.

5.2.2.2 Class CsEmMailRelA

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

Used to send email to recipients.

Attributes:

`myAttachedFiles`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
No Inheritance
Internal stored (possibly uuencoded) message bodies.

`myBCCList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
No Inheritance
Internal list of BCC recipients of the message.

`myCCList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
No Inheritance
Internal list of CC's recipients of the message.

`myHeaderList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
No Inheritance
Internal list of headers.

`myMessageBody`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
No Inheritance
Internal copy of the message body.

`myToList`

Privilege: Private
Data Type: EcTChar*
Default Value: NOT IDENTIFIED
No Inheritance
Internal list of recipients of the message.

Operations:

`EcUtStatus AddBCC (str:EcTChar*)`

Privilege: Public
No Inheritance
Will add the name string to the list of BCC'd recipients. Commas may be used to separate multiple usernames.

`EcUtStatus AddCC (str:EcTChar*)`

Privilege: Public
No Inheritance
Will add the name string to the list of CC'd recipients. Commas may be used to separate multiple usernames.

`EcUtStatus AddHeader (str:EcTChar*)`

Privilege: Public
No Inheritance
Will add an arbitrary header line to the message, the entire line you wish to appear in the header must be specified. Multiple calls will add multiple headers.

`EcUtStatus AddMessage (str:EcTChar*)`

Privilege: Public
No Inheritance
Will add the text to the currently stored body of the message. If there are attached files that have been already attached, calls to the AddMessage will add text after the file.

`EcUtStatus AddTo (str:EcTChar*)`

Privilege: Public
No Inheritance
Will add the name string to the list of recipients on the 'To:' line. Commas may be used to separate multiple usernames.

`EcUtStatus AttachFile (str:EcTChar*)`

Privilege: Public
No Inheritance
Will add the file named to the internal file storage. The file will be read in at the point of this call. The file may be uuencoded for transmittal.

`void CsEmMailRelA ()`

Privilege: Public
No Inheritance
This is the default constructor. It will not fill in any of the user-settable fields.

`EcUtStatus Send ()`

Privilege: Public
No Inheritance
Will send the message in the current state. It will not delete any of the internally stored data.

`EcUtStatus Subject (str:EcTChar*)`

Privilege: Public
No Inheritance
Will set the subject to the passed text string. Will delete any previously created subjects.

`void ~CsEmMailRelA ()`

Privilege: Public
No Inheritance
Will delete the object and all internal storage used. If mailing large files, it is recommended that this be performed as soon as possible.

5.2.2.3 Class CsFtFTPRelB

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

CsFtFTPRelB provides an API for application programmer to initiates a FTP session to transfer files non-interactively. It also provides the capability to schedule file transfer.

Attributes:

`myBatchFileB`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to store the batch script for batch mode transfer

`myConnectionOpen`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
status bit to indicate if a connection is currently open on the primary host

`myFTPRunningStatus`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
used to indicate the status of the pipe to the FTP program

`myPassword`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to store the password for the primary host to contact

`myPipe`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
pipe for sending and receiving data from the FTP client program

`myProxyConnectionOpen`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
status bit to indicate if a connection is currently open on the proxy host

`myProxyPassword`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to hold the password for the proxy connection

`myProxyRemoteHost`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to hold the hostname for the proxy connection

`myProxyStatus`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
used to hold a toggle bit to indicate if commands are directed to the primary host or the proxy host

`myProxyUserName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to hold the username for the proxy connection

`myRemoteHost`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to store the hostname for the primary machine to contact

`myTransferTimeB`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to store the time for batch mode transfer

`myUserName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
used to hold the username for the primary account

Operations:

`EcUtStatus Close ()`

Privilege: Public
No Inheritance
will close the connection

`void CsFtFTPRelB ()`

Privilege: Public
No Inheritance
constructor for session

`RWCString GetHostName (EcUtStatus)`

Privilege: Public
No Inheritance
will return the host name of the connected client

`RWCString GetLastMessage (EcUtStatus)`

Privilege: Public
No Inheritance
will return a pointer to a text buffer containing the exact text of the message that was returned by the last command. The message may span multiple lines. The buffer will be over written when the next ftp command is called with the object, so copy the data to non-volital memory if you do not plan to inspect if before executing other command.

`RWCString GetListing (EcUtStatus)`

Privilege: Public
No Inheritance
will return the file listing of the current directory on the remote mahcine

`EcTBoolean GetProxy (EcUtStatus)`

Privilege: Public
No Inheritance
will check the status of the proxy bit

`RWCString GetRemoteDirectory (EcUtStatus)`

Privilege: Public
No Inheritance
will return the working directory of the connected client

`RWCString GetUserName (EcUtStatus)`

Privilege: Public
No Inheritance
will return the name of the current user

`EcUtStatus Open ()`

Privilege: Public
No Inheritance
will open a new connection to a host and log the user on

`EcUtStatus Receive (src:RWCString,dest:RWCString)`

Privilege: Public
No Inheritance
Will transfer the named file and store it in the file named in the optional second argument. A directory may not be specified as part of a path name (calls "get" command).

`EcUtStatus ReceiveAll (RWCString)`

Privilege: Public
No Inheritance
will transfer all files on the remote machine in the current directory

`EcUtStatus ReceiveMatching (RWCString)`

Privilege: Public
No Inheritance
Will transfer all files on the remote machine in the current directory that match the specified as part of the wildcard.

`EcUtStatus ScheduleTransferB ()`

Privilege: Public
No Inheritance
will schedule the transfer

`EcUtStatus Send (src:RWCString,dest:RWCString)`

Privilege: Public
No Inheritance
Will send the named file to the remote machine and store it in the file named specified in the optional second argument. (calls "put")

`EcUtStatus SendMatching (RWCString)`

Privilege: Public
No Inheritance
Will transfer all files to the remote machine in the current directory that match the specified wildcard. A path may not be specified as part of the wildcard.

`EcUtStatus SetBatchFileB (file:RWCString)`

Privilege: Public
No Inheritance
Will set the batch script for scheduled transfer

`EcUtStatus SetFileTypeB (RWCString)`

Privilege: Public
No Inheritance
Will set the file type for transfer (binary or ASCII)

`EcUtStatus SetHostName (RWCString)`

Privilege: Public
No Inheritance
Will set the name of the host to connect to.

`EcUtStatus SetLocalDirectory (RWCString)`

Privilege: Public
No Inheritance
Will execute the lcd command to change the local working directory

`EcUtStatus SetPassword (RWCString)`

Privilege: Public
No Inheritance
Will set the password to use to connect to the remote host

`EcUtStatus SetProxy (EcTInt)`

Privilege: Public
No Inheritance
Will set or unset the "proxy" bit. This will determine if future commands are set to the primary host or the proxy host. This prevents having two seperate identical sets of functions to send commands to the two servers. If this function is not called, the default value is 0 - which means that commands will go to the primary not the proxy host.

`EcUtStatus SetRemoteDirectory (RWCString)`

Privilege: Public
No Inheritance
Will execute the cd command to change the working directory on the remote machine

`EcUtStatus SetTransferTimeB (Time:RWCString)`

Privilege: Public
No Inheritance
Will set the time for batch mode transfer

`EcUtStatus SetUserName (RWCString)`

Privilege: Public
No Inheritance
Will set the user name to be used with the FTP session

`void ~CsFtFTPRelB ()`

Privilege: Public
No Inheritance
destructor for session, will close all open connection

5.2.2.4 Class DCEAclMgr

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: ESO(Public)
Class: appServerObj(Private) creates
Class: acl_edit(Private) uses"rdacl"interface
Class: EcSeSecurity(Private) uses

Description:

This class registers a 'rdacl' interface manager object with the global DCEServer object.

Attributes:

Operations:

5.2.2.5 Class DCEAclSchema

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcSeSecurity(Private) uses

Description:

This class defines the permission bits that are available and provides a printable form of each bit and an explanatory string.

Attributes:

`_ps`

Privilege: Private
Data Type: DCESchemaPrivateState&
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents all the private state in it.

Operations:

`void AddPrintstring (ps:const DCESchemaPrintstring&)`

Privilege: Public
No Inheritance
This operation will add new print strings to a schema.

`void AddPrintstring (perm:const char* help:const char* bits:DCESchemaBitset&)`

Privilege: Public
No Inheritance
This operation will add new print strings to a schema.

`DCEAclSchema* Copy ()`

Privilege: Public
No Inheritance
This operation creates a copy of the DCEAclSchema.

`void DCEAclSchema (num_slice=1 num:int=0 ps:sec_acl_printstring_t*=0 tok:int=0)`

Privilege: Public
No Inheritance
This is the class constructor. It creates a DCEAclSchema object.

`const DCESchemaBitset& GetControlReadPermission ()`

Privilege: Public
No Inheritance
This operation will retrieve the definition of the READ control permission for the schema.

`const DCESchemaBitset& GetControlWritePermission ()`

Privilege: Public
No Inheritance
This operation will retrieve the definition of control permissions in the schema that a user must have to access the ACL information.

`char* GetPermstring (slice:int permissions:sec_acl_permset_t)`

Privilege: Public
No Inheritance
This operation returns a character string representation of permissions, either for a slice or for an entire permissions set.

`char* GetPermstring (bits:const DCESchemaBitset&)`

Privilege: Public
No Inheritance
This operation will return a character string representation of permissions, either for a slice or for an entire permissions set.

`int GetPrintstrings (slice:int len:int ps:sec_acl_printstring_t*)`

Privilege: Public
No Inheritance
This operation will fill an array ps with print string values. These are the print strings for a particular value or for a particualar schema. len is the number of elements in the ps array.

`int GetPrintstrings (len:int ps:DCESchemaPrintstring_t*)`

Privilege: Public
No Inheritance
This operation will fill an array with print string values. These are either the print strings for a particular slice, or for the entire schema. len is the number of elements in the array.

`int GetTokenizeFlag ()`

Privilege: Public
No Inheritance
This operation will return the tokenized flag.

`sec_acl_permset_t MakeBitmap (slice:int perms:const char*)`

Privilege: Public
No Inheritance
This operation will return a bit string which corresponds to the character form of permissions passed in.

`DCESchemaBitset MakeBitmap (perms:const char*)`

Privilege: Public
No Inheritance
This operation returns a bit string which corresponds to the character form of permissions passed in.

`int NumPrintstrings (slice:int)`

Privilege: Public
No Inheritance
This operation will return the number of print strings in the schema, either for a particular slice or for the entire schema.

`int NumPrintstrings ()`

Privilege: Public
No Inheritance
This operation will return the number of print strings in the schema, either for a particular slice or for the entire schema.

`int NumSlices ()`

Privilege: Public
No Inheritance
This operation will return the number of permission slices.

`sec_acl_permset_t PossibleBits (slice:int)`

Privilege: Public
No Inheritance
This operation will return the union of the permission bits that have been defined for this print string.

`const DCESchemaBitset& PossibleBits ()`

Privilege: Public
No Inheritance
This operation will return the union of the permission bits that have been defined for this print string.

`void SetControlPermissions (rw:const DCESchemaBitset&)`

Privilege: Public
No Inheritance
This operation will set the control bit that determines whether a user has access to the ACL structure. It indicates the permission bit selected for write access.

`void SetControlPermissions (r:const DCESchemaBitset& w:const DCESchemaBitset&)`

Privilege: Public
No Inheritance
This operation will set the control bit that determines whether a user has access to the ACL structure. It grants read access to view ACLs seperately from granting access to modify the ACLs.

```
void ~DCEAclSchema ()
```

Privilege: Public
No Inheritance
This is the class destructor.

5.2.2.6 Class DCEActivation

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

This abstract class is used by server developers to provide an interface onto the activation of server manager objects. Since the activation of these objects is application specific, many implementations will exist. An activation object can be registered with a Server object and be used to dynamically activate manager objects.

Attributes:

Operations:

```
void ActivateObject ()
```

Privilege: Public
No Inheritance
This is a pure virtual member function used to gain access to an implementation of activation functions. ActivateObject is called to activate the object associated with the UUID argument. The ActivateObject member function takes a DCEActivationResultT structure. Checks to see if the object state is stored in the file system, initializes the activation result and creates a new application manager object by passing the object UUID to the constructor.

5.2.2.7 Class DCEInterface

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DOF (Aggregation)

Description:

The application client class inherits from the DCEInterface class which provides the default functionality required at the client side. It includes locating a service, binding and accessing the remote objects managed by that server. Client class inherits from this DCEInterface class and as such this interface is embedded in the client class. The application programmer can modify the default behavior provided by the DCEInterface class to achieve any needed special behavior.

Attributes:

`_attempt_rebind`

Privilege: Protected Attribute
Data Type: DCERebindPolicy
Default Value: NOT IDENTIFIED
No Inheritance
rebind policy

`_auth_identity`

Privilege: Protected Attribute
Data Type: rpc_auth_identity_handle_t
Default Value: NOT IDENTIFIED
No Inheritance
Identity

`_authn_svc`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Authentication

`_authz_svc`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Authorization

`_cds_entry`

Privilege: Protected Attribute
Data Type: DCENsiObject*
Default Value: NOT IDENTIFIED
No Inheritance
CDS name

`_handle`

Privilege: Protected Attribute
Data Type: rpc_binding_handle_t
Default Value: NOT IDENTIFIED
No Inheritance
Binding handle

`_if_handle`

Privilege: Protected Attribute
Data Type: rpc_if_handle_t
Default Value: NOT IDENTIFIED
No Inheritance
DCEInterface

`_local`

Privilege: Protected Attribute
Data Type: DCEInterfaceMgr*
Default Value: NOT IDENTIFIED
No Inheritance
local object

`_object`

Privilege: Protected Attribute
Data Type: uuid_t
Default Value: NOT IDENTIFIED
No Inheritance
Object UUID

`_protection_level`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Data security

`_rebind_count`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Number of attempts to make

`_reference`

Privilege: Protected Attribute
Data Type: DCEObjectReference*
Default Value: NOT IDENTIFIED
No Inheritance
object reference

`_sec_pref_changed`

Privilege: Protected Attribute
Data Type: boolean32
Default Value: NOT IDENTIFIED
No Inheritance
Flag

`_server_cache`

Privilege: Protected Attribute
Data Type: rpc_binding_vector_t*
Default Value: NOT IDENTIFIED
No Inheritance
DCEBinding cache

`_server_principal_name`

Privilege: Protected Attribute
Data Type: unsigned char*
Default Value: NOT IDENTIFIED
No Inheritance
DCEServer principal

`_service_bound`

Privilege: Protected Attribute
Data Type: boolean32
Default Value: NOT IDENTIFIED
No Inheritance
Flag

`_set_security`

Privilege: Protected Attribute
Data Type: boolean32
Default Value: NOT IDENTIFIED
No Inheritance
Flag

Operations:

`void BindInterface ()`

Privilege: Public
No Inheritance
Used to obtain a fully bound handle from the location and binding information held in the internal state of the object. This memeber function ensures that a compatible Server object is available and is able to support the Interface and object requirements of the client object. The

BindInterface member function only needs to be used by the application code if fine grain control is requred over the binding of clients to servers. If the client object has not been bound to a server when the first RPC call is made, the BindInterface function is called has no effect if called when a valid binding exists. The Binding Interface member is virtual so that the binding policy may be overloaded in derived classes that require specialized binding mechanism.

`void DCEInterface (rpc_if_handle_t,DCEObjRefT*)`

Privilege: Public
No Inheritance
Constructor. Construct from Object reference.

`void DCEInterface (rpc_if_handle_t,DCEUuid&)`

Privilege: Public
No Inheritance
Constructor. Construct from DCEUuid. If DCEUuid& null, use the once given.

```
void DCEInterface
(rpc_if_handle_t,rpc_binding_vector_t*,DCEUuid&)
```

Privilege: Public
No Inheritance
Constructor. Construct from binding vector and DCEUuid

```
void DCEInterface (rpc_if_handle_t,unsigned char*,unsigned
char*,DCEUuid&)
```

Privilege: Public
No Inheritance
Constructor. Construct from hostname/protocol

`void DCEInterface (rpc_if_handle_t,DCENsiObject*,DCEUuid&)`

Privilege: Public
No Inheritance
Constructor. Construct from DCENsiObject

```
void DCEInterface (rpc_if_handle_t,unsigned
char*,unsigned32,DCEUuid&)
```

Privilege: Public
No Inheritance
Constructor. Construct from unsigned string

```
void DCEInterface
(rpc_if_handle_t,rpc_binding_handle_t,DCEUuid&)
```

Privilege: Public
No Inheritance
Constructor. Construct from a binding

```
unsigned32 GetAuthenticationService ()
```

Privilege: Public
No Inheritance
Used to get the authentication service type required by the client to perform authentication.

```
unsigned32 GetAuthorizationService ()
```

Privilege: Public
No Inheritance
Used to get the client side identity for the communication with a Server object.

```
rpc_binding_handle_t GetBinding ()
```

Privilege: Public
No Inheritance
Returns the current binding information for the interface object. It returns information from internal state and as such the return value is marked as const.

```
rpc_auth_identity_handle_t GetIdentity ()
```

Privilege: Public
No Inheritance
Used to get the client side identity for the communication with a Server object.

```
unsigned32 GetProtectLevel ()
```

Privilege: Public
No Inheritance
Used to get required data protection levels between the client and the server.

```
unsigned char* GetServerPrincipal ()
```

Privilege: Public
No Inheritance
Used to get the principal nameof the server object.

`void RebindInterface ()`

Privilege: Public
No Inheritance
Rebinds the binding information and perform checks that the Server object can support the client object requirements.

`void ResetBinding ()`

Privilege: Public
No Inheritance
Resets endpoint information contained in the binding handle. The client object will re-bind the endpoint information on the next RPC call.

```
void SetAuthInfo (unsigned
char*,unsigned32,unsigned32,rpc_auth_identity_handle_t,unsi
gned32)
```

Privilege: Public
No Inheritance
Used to set the client side security preferences.

`void SetAuthenticationService (unsigned32)`

Privilege: Public
No Inheritance
Set the authentication service type required by the client to perform authentication.

`void SetAuthorizationService (unsigned32)`

Privilege: Public
No Inheritance
Used to set the client side identity for the communication with a Server object.

`void SetBinding (rpc_binding_handle_t)`

Privilege: Public
No Inheritance
Resets the binding information held in the internal state of the object. Any existing binding information is removed and new binding checks are performed on the next RPC call. It takes a binding handle as an argument.

`void SetBinding (DCEBinding&)`

Privilege: Public
No Inheritance
Resets the binding information held in the internal state of the object. Any existing binding information is removed and new binding checks are performed on the next RPC call. It takes a DCEBinding as an argument.

`void SetBinding (unsigned char*)`

Privilege: Public
No Inheritance
Resets the binding information held in the internal state of the object. Any existing binding information is removed and new binding checks are performed on the next RPC call. It takes an string as argument.

`void SetIdentity (rpc_auth_identity_hanlde_t)`

Privilege: Public
No Inheritance
Used to set the client side identity for the communication with a Server object.

`void SetProtectLevel (unsigned32)`

Privilege: Public
No Inheritance
Used to set required data protection levels between the client and the server.

`void SetRebind (DCERebindPolicy)`

Privilege: Public
No Inheritance
Sets the rebind policy of the interface object. By default the interface object has the rebind policy of never_rebind. The rebind policy allows the specification of a re-bind policy for client objects. If communication with a Server object results in a failure the client object can be instructed to attempt a re-bind. The rebind policy can be: 1.never_rebind:No rebind is done, failure is passed reported to caller as an exception.
2.attempt_rebind:Client object will attempt to rebind and replay the failed operation. The default re-bind algorithm is to first reset the endpoint on the binding handle used. If this fails then an attempt is made to obtain a new binding (which could mean going to a totally different Server object). Re-bind is attempted once if the rebind or the re-playing of the failed operation presents another failure and an exception is passed to the client. 3.wait_on_rebind: Does the same as the

attempt_rebind option, however this option will continue to retry a re-bind until it is successful or the calling thread is terminated. It takes as argument the DCERebindPolicy.

```
void SetRebind (DCERebindPolicy,unsigned32)
```

Privilege: Public
No Inheritance
Sets the rebind policy of the interface object. By default the interface object has the rebind policy of never_rebind. The rebind policy allows the specification of a re-bind policy for client objects. If communication with a Server object results in a failure the client object can be instructed to attempt a re-bind. The rebind policy can be: 1.never_rebind:No rebind is done, failure is passed reported to caller as an exception. 2.attempt_rebind:Client object will attempt to rebind and replay the failed operation. The default re-bind algorithm is to first reset the endpoint on the binding handle used. If this fails then an attempt is made to obtain a new binding (which could mean going to a totally different Server object). Re-bind is attempted once if the rebind or the re-playing of the failed operation presents another failure and an exception is passed to the client. 3.wait_on_rebind: Does the same as the attempt_rebind option, however this option will continue to retry a re-bind until it is successful or the calling thread is terminated. It takes as arguments the DCERebindPolicy and an integer.

```
void SetServerName (unsigned char*, unsigned32 =
rpc_c_ns_syntax_default)
```

Privilege: Public
No Inheritance
Set the CDS entry name that is used to locate the required Server object. When SetServerName is called, the existing binding handle is invalidated and the client object will rebind using the new server name on the next RPC call.

```
void SetServerObject (DCEUuid&)
```

Privilege: Public
No Inheritance
Sets or modifies the DCE object uuid of a client binding. Checks are made to ensure the Server Object supports the required DCE object uuid.

`void SetServerPrincipal (unsigned char*)`

Privilege: Public
No Inheritance
Used to set the principal name of the server object.

`void ~DCEInterface ()`

Privilege: Public
No Inheritance
Destructor

5.2.2.8 Class DCEInterfaceMgr

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DOF (Aggregation)

Description:

There can be multiple implementations for a given interface. The application server implementation class inherits from the InterfaceMgr class and the DCEObj class. The DCEInterfaceMgr class is the base abstract class from which all generated server side classes are derived. This class allows the application programmer to define and attach a RefMon class where the desired level of security (authentication) can be specified. The DCEObj class provides the concept of the Generic object which can have multiple interfaces. This class provides the functionality to associate a RefMon object to all the interfaces that it is associated to. Server implementation class inherits from these classes and as such their interfaces are embedded in the server class. The application programmer can modify the default behavior provided by these classes to achieve any needed special behavior.

Attributes:

`_if_impl_mgr`

Privilege: Protected Attribute
Data Type: rpc_mgr_epv_t
Default Value: NOT IDENTIFIED
No Inheritance
C Manager epv for DCE

Operations:

```
void DCEInterfaceMgr (:rpc_if_handle_t :uuid_t*
:rpc_mgr_epv_t)
```

Privilege: Public
No Inheritance
This constructor creates a DCEInterface object with the interface identifier.

`void DCEInterfaceMgr (:rpc_if_handle_t :uuid_t* :uuid_t* :rpc_mgr_epv_t)`

Privilege: Public
No Inheritance
This constructor will create DCEInterfaceMgr object with the interface and the object identifier.

`void DCEInterfaceMgr (:rpc_if_handle_t :DCEObj& :uuid_t* :rpc_mgr_epv_t)`

Privilege: Protected Operation
No Inheritance
This constructor creates DCEInterfaceMgr object with the object reference.

`rpc_mgr_epv_t GetEpv ()`

Privilege: Public
No Inheritance
Used to access the endpoint vector for the DCEInterfaceMgr object. Access to this data structure is required by the class library but not normally needed for general development.

`DCEUuid& GetObj ()`

Privilege: Public
No Inheritance
Provides access to the DCE object identifier information for the DCEInterfaceMgr object.

`DCEUuid& GetType ()`

Privilege: Public
No Inheritance
Provides access to the DCE implementation type information for the DCEInterfaceMgr object.

`void SetRefMon (rm:DCERefMon*)`

Privilege: Public
No Inheritance
Set reference monitor

`void ~DCEInterfaceMgr ()`

Privilege: Public
No Inheritance
This is the destructor. It will deallocate memory occupied by DCEInterfaceMgr object.

5.2.2.9 Class DCEObj

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DOF (Aggregation)

Description:

This class provides the concept of DCE object which is a logical entity and can have multiple interfaces. This class is used to collect related interfaces together to form a DCE object. Object of this class can be registered with a Server object.

Attributes:

Operations:

```
void DCEObj (id:uuid_t*,rm:DCERefMon*,act:DCEActivityBase*)
```

Privilege: Public
No Inheritance
Constructor.

```
DCEUuid& GetId ()
```

Privilege: Public
No Inheritance
Obtains the object UUID associated with the DCEObj class instance.

```
DCEInterfaces& GetInterfaces ()
```

Privilege: Public
No Inheritance
Obtains a list of the interfaces supported by the manager object.

```
DCEObjectReference& GetObjectReference ()
```

Privilege: Public
No Inheritance
Obtains an object reference fro the DCEObj class instance. This reference can be used by clients to create a binding to the server base object.

```
void RegisterInterface (DCEInterfaceMgr*)
```

Privilege: Public
No Inheritance
Registers the interface data with the DCEObj class. This class maintains a list of all interfaces supported by the manager object associated with the DCEObj class.

`void SetAllActivity (DCEActivityBase*)`

Privilege: Public
No Inheritance
Sets the activity object for all of the DCEInterfaceMgr objects registered with DCEObj. This allows multiple objects to share the same activity object and counter.

`void SetAllRefMon ()`

Privilege: Public
No Inheritance
Used to associate a RefMon object with all of the DCEInterfaceMgr objects registered with the DCEOjec object. Therefore, all interfaces supported by the manager object share the same RefMon object.

`void SetAllRefMon (DCERefMon*)`

Privilege: Public
No Inheritance
Set reference monitor

`void ~DCEObj ()`

Privilege: Public
No Inheritance
Destructor.

5.2.2.10 Class DCEPassword

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

This class supports implementation of password storage and access.

Attributes:

Operations:

5.2.2.11 Class DCERefMon

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: appServerObj(Private) creates
Class: ESO(Public) registerswithappSrvObject

Description:

This class provides an abstraction of a reference monitor that controls the client object's access to a manager object.

Attributes:

Operations:

5.2.2.12 Class DCESecId

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcSeSecurity(Private) uses

Description:

This utility class encapsulates the sec_id_t data type of DCE.

Attributes:

Operations:

5.2.2.13 Class DCEUuid

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: appServerObj(Private) creates

Description:

This utility class encapsulates the DCE data type uuid_t. It provides a flexible way of dealing with the various forms of representing a UUID.

Attributes:

Operations:

5.2.2.14 Class ECSAcl

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcSeSecurity(Private) uses

Description:

This class is used to access a DCE access control list. It maintains all the information about an ACL. The ECSAclDb stores Acl information, and makes it visible for reading through this interface. The ECSModifyableAcl interface is used for editing ACLs. A ECSAcl is a read-only object. In order to change a ECSAcl, it must be either replaced completely with another ECSAcl, or converted into a ECSModifyableAcl in order to be edited through an application interface.

Attributes:

Operations:

`void ECSAcl ()`

Privilege: Public
No Inheritance
This is the default constructor. It creates a ECSAcl object.

`DCESchemaBitset GetAccess ()`

Privilege: Public
No Inheritance
This operation returns the set of permissions that can be granted to the current client by the ECSAcl.

`DCESecId GetGroupObj ()`

Privilege: Public
No Inheritance
This operation will return the group owner identity. The same explanation applies as for GetUserObj().

`DCESecId GetUserObj ()`

Privilege: Public
No Inheritance
This operation will return the user owner-identity associated with the ECSAcl. It is a convenience routine because the information might actually be stored in the database, in which case the ECSAcl needs to get the information from the database.

`boolean32 IsAuth (desired_perms:DCESchemaBitset)`

Privilege: Public
No Inheritance
This operation is the most important member function that ECSAcl provides. It returns TRUE if the ECSAcl will grant the desired permissions to the client. It returns FALSE if all the desired permissions can not be granted.

```
boolean32 IsAuth (desired_perms:DCESchemaBitset
identity:const sec_id_pac_t*)
```

Privilege: Public
No Inheritance
This operation checks the authorization rights of a third party whose identity is passed to the function as the second argument.

```
void ReleaseAcl ()
```

Privilege: Public
No Inheritance
This operation advises the system that the ECSAcl is no longer needed. In the implementation to be provided, a call to this function unlocks the ECSAcl so that other clients may use it.

```
sec_acl_t* Slice ()
```

Privilege: Public
No Inheritance
This operation will return the DCE data structure sec_acl_t. It extracts just one slice of permissions since that is all that a sec_acl_t will hold. This conversion from an ECSAcl object to a sec_acl is needed by the 'rdacl' interface.

```
void ~ECSAcl ()
```

Privilege: Public
No Inheritance
This is the destructor. It deletes a ECSAcl object.

5.2.2.15 Class ECSAclDb

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcSeSecurity(Private) uses

Description:

This class defines the interface to the ACL database. The ACL databases store and retrieve all ACLs based on an object name. ACLs can be organized and retrieved in different ways such as by ACL type (object, default object, or default container), and according to the permission slice. Each database is associated with only one DCEAclSchema that may contain multiple slices. The ECSAclDb implementation must define the ACL locking protocol and provide the mechanism to use it.

Attributes:

Operations:

```
ECSModifyableAcl* CreateAcl (object_name:const char*
newuser_obj:const DCESecId=NULL newgroup_obj:const
DCESecId*=NULL
sec_acl_type:sec_acl_type_t=sec_acl_type_object)
```

Privilege: Public
No Inheritance
This operation will create an ECSModifyableAcl object that is brand new; that is, there was no ECSAcl in this database corresponding to the obejct name and ECSAcl type supplied. It creates a blank template that can be filled in and written back to the database using the ECSModifyableAcl member functions. The owner and the group identities may be specified here or may be filled in later by the member functions of ECSModifyableAcl.

```
void DeleteAcl (object_name:const char*
sec_acl_type:sec_acl_type_t=sec_acl_type_object)
```

Privilege: Public
No Inheritance
This operation will delete an ECSAcl from the database given the object name and an optional ACL type.

```
void ECSAclDb ()
```

Privilege: Public
No Inheritance
This is the default constructor.

```
char* GetDbName ()
```

Privilege: Public
No Inheritance
This operation will return the name of the ECSAclDb database in character string format.

```
DCESecId GetGroupObj (object_name:const char*)
```

Privilege: Public
No Inheritance
This operation will return a DCESecId object that represents the group owner of the object; that is, the identity of the group that will match on the group_obj ACL entry.

```
ECSModifyableAcl* GetModifyableAcl (object_name:const char*
sec_acl_type:sec_acl_type_t=sec_acl_type_object
new_acl:sec_acl_t=NULL)
```

Privilege: Public
No Inheritance
This operation should be called when users want to edit an Acl in an application interface rather than through the 'rdacl' interface. It creates a ECSModifyableAcl object containing the current existing Acl information for the specified object in the database. The changes made to the ECSModifyableAcl are not used in authorization decisions until they are explicitly committed back to the database.

```
const DCEAclSchema* GetSchema ()
```

Privilege: Public
No Inheritance
This operation will return a pointer to the schema object that describes the permission bits managed by this database.

```
DCESecId GetUserObj (object_name:const char*)
```

Privilege: Public
No Inheritance
This operation returns a DCESecId object that represents the user owner of the object; that is, the identity of the principal that will match on the user_obj ACL entry.

```
boolean32 IsAuth (object_name:const char*
desired_perms:DCESchemaBitset)
```

Privilege: Public
No Inheritance
This operation make an authorization decision. It returns a TRUE if the caller has the desired_perms and a FALSE otherwise. The caller's information is obtained from within the implementation of this member function supplied by the DCEClientInfo object that is obtained when the server is invoked either through the application interface or thropugh the 'rdacl' interface.

```
boolean32 IsAuth (objecvt_name:const char*
desired_perms:DCESchemaBitset identity:const sec_id_pac_t*)
```

Privilege: Public
No Inheritance
This operation is similar to the first form but accepts a string representation of the permission set desired instead of a DCESchemaBitset.

```
boolean32 IsAuth (object_name:const char* perm_string:const
char*)
```

Privilege: Public
No Inheritance
This is similar to the first and second form except that it takes a DCE PAC structure to identify a third party. If the requestor and that third party have the desired permissions, it returns a TRUE. This form is needed to implement rdacl_get_access_on_behalf.

```
ECSAcl* Lookup (object_name:const char*
sec_acl_type:sec_acl_type_t=sec_acl_type_object)
```

Privilege: Public
No Inheritance
This operation will lookup an ECSAcl in the database based on the object name and the type of ACL. It returns a pointer to an ECSAcl. A ECSAcl is read-only and none of the member functions can modify its state.

```
void ~ECSAclDb ()
```

Privilege: Public
No Inheritance
Destructor

5.2.2.16 Class ECSAclStorageManager

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcSeSecurity(Private) uses

Description:

This class supports multiple ACL databases. It manages the ACL databases being used by a server, providing registration and search services for these databases. It provides manager type and schema information as needed by the rdacl interface. It also provides functionality for creating a new ECSAclDb.

Attributes:

Operations:

```
ECSAclDb CreateNewDatabase (database_name:const char*
schema:const DCEAclSchema* persistent_name:const char*=0)
```

Privilege: Public
No Inheritance
This operation creates a new DCEAclDb object with the given name and

associated with the given schema. The third optional parameter allows the database implementor to provide a constructor that uses the full pathname of a persistent datastore.

`void DoneWithDb (:ECSAclDb*)`

Privilege: Public
No Inheritance
This operation will indicate when done with the specified database.

`void ECSAclStorageManager ()`

Privilege: Public
No Inheritance
This is the class default constructor. It creates an ECSAclStorageManager object.

`ECSAclDb GetDb (database_name:const char*)`

Privilege: Public
No Inheritance
This operation provides a database handle from the DCEAclStorageMgr by providing the database name.

`ECSAclDb GetDb (manager_type:const DCEMgrType*)`

Privilege: Public
No Inheritance
This operation provides a database handle from the DCEAclStorageMgr by providing a manager type.

`const char* GetDbName (manager_type:const DCEMgrType*)`

Privilege: Public
No Inheritance
This operation returns the name of the database corresponding to the manager type passed in. This is used in constructing the print string for the manager type required by rdacl_get_printstrings.

```
void GetManagerTypes (component_name:const char*
sec_acl_type:sec_acl_type_t size_avail:unsigned32
size_used:unsigned32* num_types:unsigned32*)
```

Privilege: Public
No Inheritance
This operation will return the array of DCEMgrType corresponding to the databases that contain the ACLs protecting the object identified by the first parameter. It supports the implementation of rdacl_get_manager_types and rdacl_get_manager_types_semantics.

```
const DCEAclSchema GetSchema (manager_type:DCEMgrType*
index:int* manager_type_chain:DCEMgrType*)
```

Privilege: Public
No Inheritance
This operation is convenient for implementing the 'rdacl' interface, and can be ignored by most users. It returns a pointer to a DCEAclSchema object for the database identified by the first parameter, which is manager type. In this case, the manager type identifies a particular slice of permissions.

```
void Register (database:ECSAclDb*)
```

Privilege: Public
No Inheritance
This operation allows the server developer to construct an ECSAclDb and then to register it directly with the ECSAclStorageManager object.

```
void ~ECSAclStorageManager ()
```

Privilege: Public
No Inheritance
This is the class destructor. It frees the memory occupied by the ECSAclStorageManager object.

5.2.2.17 Class ECSModifyableAcl

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcSeSecurity(Private) uses

Description:

This is a temporary copy of ECSAcl that can be used for editing directly by the server or through an application-defined interface. Users of the 'rdacl' interface do not use ECSModifyableAcl directly, but rather ECSModifyableAcl provides an alternative path for editing ACLs. The contents of a ECSModifyableAcl are not seen when authorization decisions are made. The changes don't take effect until the ECSModifyableAcl is committed back to the database, at which time the ECSModifyableAcl itself no longer exists.

Attributes:

Operations:

`void AddAclEntry (permissions:DCESchemaBitset acl_entry_type:sec_acl_entry_type_t pgo_name:DCESecId*=NUL)`

Privilege: Public
No Inheritance
This operation can be used to add new ACL entries, once a ECSModifyableAcl object has been created. It takes permissions, the acl_entry_type, and the optional pgo_name identity corresponding to the ACL entry.

`void CommitAcl ()`

Privilege: Public
No Inheritance
This operation will write the ECSModifyableAcl contents into the database and deletes the ECSModifyableAcl. When CommitAcl() completes, authorization decisions will take the new ACL into account.

`void ECSModifyableAcl ()`

Privilege: Public
No Inheritance
This is the class default constructor.

`void SetGroupObj (identity:DCESecId&)`

Privilege: Public
No Inheritance
This operation will allow the caller to change the identity of the group owner of the object protected by this ACL.

`void SetUserObj (identity:DCESecId&)`

Privilege: Public
No Inheritance
This operation allows the caller to change the identity of the user owner of the object protected by this ACL.

`void ~ECSModifyableAcl ()`

Privilege: Public
No Inheritance
This is the class destructor.

5.2.2.18 Class ESO

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: DCEAclMgr(Public)
Class: appServerObj(Private) registerswithCDS
Class: DCERefMon(Public) registerswithappSrvObject
DOF (Aggregation)

Description:

A generic Server class is provided at the server side. This class deals with the management of the objects that implement the interfaces. This class provides the functionality to interact with the objects that implement the interfaces. This class provides the functionality to interact with the naming and security services on behalf of the objects it manages. There will only be one instance of this class in a process. An instance of this will be created and bound to a global variable called theServer. Server main application uses this class to manage the objects. The server main is a driver program which uses this class to create a deamon running all the time listening to incoming requests. Each time a request arrives, this application decides which server manager should address the request and spawns a new thread to run that implementaion of the server application. Each server driver can manage several server managers. A server manager is one implementaion of the server. This class provides all the needed interfaces to deal with core servies like the naming and the security. This class provides an abstraction of the interfaces to these core services, which the application programmer can use. The actual interface between this service and the core services is private and the application programmer never uses that directly. The number of requests a server main can take are limited, and can be set by the application programmer. It also maintains a queue to keep the incoming request if all the server implementaions are busy. The length of the queue is configurable by the application programmer. If a request arrives when the queue is full, then the request is ignored without any notifications. Application specific behavior can be extended by defining another object inheriting from the global server object (GSO). For example, if the application need to register it some local database, the register method can be implemented in the newly created global server object. This can be done by application programmer.

Attributes:

`_Init`

Privilege: Protected Attribute
Data Type: int
Default Value: NOT IDENTIFIED
No Inheritance
One time init flag

`_activator`

Privilege: Protected Attribute
Data Type: DCEActivation*
Default Value: NOT IDENTIFIED
No Inheritance
Activation object (optional)

`_auth_arg_val`

Privilege: Protected Attribute
Data Type: void*
Default Value: NOT IDENTIFIED
No Inheritance
KeyTab file arg

`_auth_obj`

Privilege: Protected Attribute
Data Type: DCEMgmtAuthorizer*
Default Value: NOT IDENTIFIED
No Inheritance
Authorizer for management functions

`_authentication_service_type`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Authn server type

`_description`

Privilege: Protected Attribute
Data Type: char*
Default Value: NOT IDENTIFIED
No Inheritance
Text description

`_endpoints_registered`

Privilege: Protected Attribute
Data Type: boolean
Default Value: NOT IDENTIFIED
No Inheritance
Flag, are the endpoints registered

`_exports`

Privilege: Protected Attribute
Data Type: DCEInterfaceList
Default Value: NOT IDENTIFIED
No Inheritance
List of Interfaces for export

`_free_list`

Privilege: Protected Attribute
Data Type: DCEInterfaceList
Default Value: NOT IDENTIFIED
No Inheritance
List of Interfaces for deletion

`_group`

Privilege: Protected Attribute
Data Type: DCENsiGroup*
Default Value: NOT IDENTIFIED
No Inheritance
Optional CDS group

`_interface_registered`

Privilege: Protected Attribute
Data Type: boolean
Default Value: NOT IDENTIFIED
No Inheritance
Flag, are interfaces registered with runtime

`_interfaces`

Privilege: Protected Attribute
Data Type: DCEInterfaceList
Default Value: NOT IDENTIFIED
No Inheritance
List of interfaces

`_key_retrieval_obj`

Privilege: Protected Attribute
Data Type: DCEKeyRetriever
Default Value: NOT IDENTIFIED
No Inheritance
Key retriever object

`_max_call_requests`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Maximum number of concurrent calls

`_objects`

Privilege: Protected Attribute
Data Type: DCEObjectList
Default Value: NOT IDENTIFIED
No Inheritance
List of server objects

`_profile`

Privilege: Protected Attribute
Data Type: DCENsiProfile *
Default Value: NOT IDENTIFIED
No Inheritance
Optional CDS profile

`_profile_prio`

Privilege: Protected Attribute
Data Type: unsigned32
Default Value: NOT IDENTIFIED
No Inheritance
Profile priority

`_protocols`

Privilege: Protected Attribute
Data Type: DCEProtocolList
Default Value: NOT IDENTIFIED
No Inheritance
List of protocol sequences

`_protocols_registered`

Privilege: Protected Attribute
Data Type: boolean
Default Value: NOT IDENTIFIED
No Inheritance
Flag, are protocols registered

`_reaper`

Privilege: Protected Attribute
Data Type: DCEPthread*
Default Value: NOT IDENTIFIED
No Inheritance
Cleanup thread

`_servent`

Privilege: Protected Attribute
Data Type: DCENsiEnry*
Default Value: NOT IDENTIFIED
No Inheritance
CDS server entry

`_server_is_listening`

Privilege: Protected Attribute
Data Type: boolean
Default Value: NOT IDENTIFIED
No Inheritance
Flag, is server object listen loop

`_server_is_registered`

Privilege: Protected Attribute
Data Type: boolean
Default Value: NOT IDENTIFIED
No Inheritance
Flag, is server registered in CDS

`_server_lock`

Privilege: Protected Attribute
Data Type: pthread_mutex_t
Default Value: NOT IDENTIFIED
No Inheritance
Main Mutex for DCEServer class

`_server_principal_name`

Privilege: Protected Attribute
Data Type: unsigned char*
Default Value: NOT IDENTIFIED
No Inheritance
Principal name

`_use_protocols`

Privilege: Protected Attribute
Data Type: DCEProtSeqReg
Default Value: NOT IDENTIFIED
No Inheritance
Protocol registration policy

Operations:

`void ExportObject (uuid_t*)`

Privilege: Public
No Inheritance
Exports the object information into the CDS.

`DCEActivation* GetActivationObject ()`

Privilege: Public
No Inheritance
Used to access the Activation object associated with the server.

`void* GetAuthArg ()`

Privilege: Public
No Inheritance
The member functions defined above can be used to specify optional parameter information that will be passed to the key retrieval mechanism when keys are requested. Security keys are stored in a key table, the default key table is a simple file with a well known filename. If the Server object is using the default key retrieval machanism but the key is not stored in the default key table the above functions may be used to set the filename of the key table to use. If a KeyRetriever object is registered with the server object, additional argument information can be passed to the KeyRetriever object by using the SetAuthArg function. The interpretation of the information set by the SetAuthArg member function is dependent on the value of the authentication service type.

`unsigned char* GetAuthnService ()`

Privilege: Public
No Inheritance
Gets the authentication service type used to do the authentication between the Server object and the client.

`char* GetDescription ()`

Privilege: Public
No Inheritance
Used to get a textual description for the server object.

`DCENsiEntry* GetEntry ()`

Privilege: Public
No Inheritance
Used to obtain the CDS class for the server objects entry name.

`DCEExportScope GetExportScope ()`

Privilege: Public
No Inheritance
Gets the export policy that controls the registration of interface and object information with a Server object. By default, Interface and object information is registered with the local RPC runtime and the endpoint mapper.

`const char* GetGroupName ()`

Privilege: Public
No Inheritance
Gets the group name of the entry from the CDS.

`DCEHostPolicy GetHostPolicy ()`

Privilege: Public
No Inheritance
Gets the per host operation policy of the Server object. By default multiple instances of a particular server object can exist per host. This allows the option of enforcing a single instance per host.

`DCEKeyRetriever* GetKeyRetriever ()`

Privilege: Public
No Inheritance
Gets the key retriever object associated with the server. This is used when an application programmer wants to run a server with an identity different from the principal starting that server.

`DCEMgmtAuthorizer* GetManagementAuthorizer ()`

Privilege: Public
No Inheritance
Gets a MgmtAuthorizer object for a Server object. If the Server Object has no MgmtAuthorizer object, then the default DCE policy is used to control access to the server management functions. However, if such an object is registered it will be used to control access to management functionality.

`char* GetName ()`

Privilege: Public
No Inheritance
Gets the server entry name from the CDS. Cell relative or a global names can be used to denote a server entry.

`unsigned char* GetPrincipalName ()`

Privilege: Public
No Inheritance
Gets the security principal's name of the Server object.

`DCENsiProfile* GetProfile ()`

Privilege: Public
No Inheritance
Returns a CDS profile name.

`char* GetProfileName ()`

Privilege: Public
No Inheritance
Retrieves profile information from the CDS.

`unsigned32 GetProfilePriority ()`

Privilege: Public
No Inheritance
Used to retrieve profile priority information from the CDS.

`void Listen ()`

Privilege: Public
No Inheritance
Keeps the server in the listen state waiting for incoming requests. Maximum number of concurrent calls can be optionally specified here.

`void RegisterObject ()`

Privilege: Public
No Inheritance
Registers a new object with the server and optionally with the CDS. The server must have at least one object registered before the Listen method can be called.

`void RemoveObjects ()`

Privilege: Public
No Inheritance
Server manager objects that are unregistered from the server object may be placed on an internal list. The RemoveObjects function can be used to free up resources associated with the manager objects. This function is called automatically by a garbage collection thread.

`static pthread_addr_t ServerCleanup (void*)`

Privilege: Public
No Inheritance
Implements the cleanup handler for the server when the server goes down like removing binding information from the CDS.

`boolean ServerIsListening ()`

Privilege: Public
No Inheritance
Returns true if the Server object is currently in the listening state and false otherwise.

`void SetActivationObject (DCEActivation*,DCEUuidVector&)`

Privilege: Public
No Inheritance
SetActivationObject is used to register an Activation object with the server. The activation object can be associated with a single interface or a vector of interfaces.

`void SetActivationObject (DCEActivation*,DCEUuid&)`

Privilege: Public
No Inheritance
SetActivationObject is used to register an Activation object with the server. The activation object can be associated with a single interface or a vector of interfaces.

```
void* SetAuthArg (arg:void*)
```

Privilege: Public
No Inheritance
The member functions defined above can be used to specify optional parameter information that will be passed to the key retrieval mechanism when keys are requested. Security keys are stored in a key table, the default key table is a simple file with a well known filename. If the Server object is using the default key retrieval machanism but the key is not stored in the default key table the above functions may be used to set the filename of the key table to use. If a KeyRetriever object is registered with the server object, additional argument information can be passed to the KeyRetriever object by using the SetAuthArg function. The interpretation of the information set by the SetAuthArg member function is dependent on the value of the authentication service type.

```
void SetAuthInfo (unsigned char*, unsigned32,
DCEKeyRetriever*, void*)
```

Privilege: Public
No Inheritance
Registers the principal name and authorization (optionally a key retriever) information with the runtime RPC. Two versions of the SetAuthInfo member function are provided. The first one allows a KeyRetriever object to be set for the Server object allowing an alternative mechanism for retrieving security keys to be used. The second version, just allows the setting of principal name and authentication service type. In this case the default key retrieval mechanism provided by DCE is used.

```
void SetAuthInfo (unsigned char *, unsigned32, void*)
```

Privilege: Public
No Inheritance
Registers the principal name and authorization (optionally a key retriever) information with the runtime RPC. Two versions of the SetAuthInfo member function are provided. The first one allows a KeyRetriever object to be set for the Server object allowing an alternative mechanism for retrieving security keys to be used. The second version, just allows the setting of principal name and authentication service type. In this case the default key retrieval mechanism provided by DCE is used.

`void SetAuthnService (this_service:unsigned32)`

Privilege: Public
No Inheritance
Sets the authentication service type used to do the authentication between the Server object and the client.

`void SetDescription (desc:char*)`

Privilege: Public
No Inheritance
Used to obtain the name syntax of the server if specified.

`void SetExportScope (scope:DCEExportScope)`

Privilege: Public
No Inheritance
Sets the export policy that controls the registration of interface and object information with a Server object. By default, Interface and object information is registered with the local RPC runtime and the endpoint mapper.

`void SetGroupName (cons DCENsiName&)`

Privilege: Public
No Inheritance
Sets the group name of the entry from the CDS.

`void SetGroupName (const unsigned char*, const unsigned32)`

Privilege: Public
No Inheritance
Sets the group name of the entry from the CDS.

`void SetGroupName (const char*,const unsigned32)`

Privilege: Public
No Inheritance
Sets the group name of the entry from the CDS.

`void SetHostPolicy (policy:const DCEHostPolicy)`

Privilege: Public
No Inheritance
Sets the per host operation policy of the Server object. By default multiple instances of a particular server object can exist per host. This allows the option of enforcing a single instance per host.

```
void SetKeyRetriever (kr:DCEKeyRetriever*)
```

Privilege: Public
No Inheritance
Sets the key retriever object associated with the server. This is used when an application programmer wants to run a server with an identity different from the principal starting that server.

```
void SetManagementAuthorizer (ma:DCEMgmtAuthorizer*)
```

Privilege: Public
No Inheritance
Sets a MgmtAuthorizer object for a Server object. If the Server Object has no MgmtAuthorizer object, then the default DCE policy is used to control access to the server management functions. However, if such an object is registered it will be used to control access to management functionality.

```
void SetName (const char*,const unsigned32)
```

Privilege: Public
No Inheritance
Sets the server entry name from the CDS. Cell relative or a global name can be used to denote a server entry.

```
void SetName (const unsigned char*,const unsigned32)
```

Privilege: Public
No Inheritance
Sets the server entry name from the CDS. Cell relative or a global name can be used to denote a server entry.

```
void SetPrincipalName (unsigned char*)
```

Privilege: Public
No Inheritance
Sets the security principal's name of the Server object.

```
void SetProfileName (const char*, const unsigned32)
```

Privilege: Public
No Inheritance
Registers a profile entry in the CDS.

```
void SetProfileName (const unsigned char*,const unsigned32,)
```

Privilege: Public
No Inheritance
Registers a profile entry in the CDS.

`void SetProfileName (const DCENsiName&)`

Privilege: Public
No Inheritance
Registers a profile entry in the CDS.

`void SetProfilePriority (prio:unsigned32)`

Privilege: Public
No Inheritance
Used to manipulate the CDS profile information. Specifically, used to set profile priority information, or to associate a priority to the entry.

`void Shutdown ()`

Privilege: Public
No Inheritance
Stops the server for new requests, unregisters the information from the CDS.

`void Stop ()`

Privilege: Public
No Inheritance
Causes the server to stop listening for new incoming requests.

`void UnExportInterface (DCEInterfaceMgr&)`

Privilege: Public
No Inheritance
Unexports the server interface information from the CDS.

`void UnExportObject (uuid_t*)`

Privilege: Public
No Inheritance
Removes the object information from the CDS.

`void UnRegisterObject (DCEInterfaceMgr&, boolean32)`

Privilege: Public
No Inheritance
Unregisters a new object with the server and optionally with the CDS. The server must have at least one object registered before the Listen method can be called.

`void UnRegisterObject (uuid_t*, boolean32)`

Privilege: Public
No Inheritance
Unregisters a new object with the server and optionally with the CDS. The server must have at least one object registered before the Listen method can be called.

`void UnRegisterObject (DCEObj&, boolean32)`

Privilege: Public
No Inheritance
Unregisters a new object with the server and optionally with the CDS. The server must have at least one object registered before the Listen method can be called.

`void UseAllProtocols (const unsigned32)`

Privilege: Public
No Inheritance
Uses all the protocols and sets the maximum number of request that can be handled by the server object.

`void UseAllProtocols (DCEInterfaceMgr&, const unsgined32)`

Privilege: Public
No Inheritance
Uses all the protocols and sets the maximum number of request that can be handled by the server object.

`void UseProtocol (const unsigned char*,const unsigned32)`

Privilege: Public
No Inheritance
Instructs the server to use a particular protocol for client requests. This can be called multiple times to set multiple protocols.

`void UseProtocol (const unsigned char*,const unsigned char*,const unsigned32)`

Privilege: Public
No Inheritance
Instructs the server to use a particular protocol for client requests. This can be called multiple times to set multiple protocols.

`void UseProtocol (const unsigned char*,DCEInterfaceMgr&,const unsigned32)`

Privilege: Public
No Inheritance
Instructs the server to use a particular protocol for client requests. This can be called multiple times to set multiple protocols.

`boolean32 _AuthFunc (handle_t,unsigned32,unsigned32*)`

Privilege: Public
No Inheritance
Protected member function.

`DCEInterfaceList* _GetInterfaces ()`

Privilege: Public
No Inheritance
Returns a pointer to a list of each unique interface provided by the objects.

`void _GetKey (void*,unsigned_char_p_t,unsigned32,void**,unsigned32*)`

Privilege: Public
No Inheritance
Protected member function.

`DCEObjectList* _GetObjectList ()`

Privilege: Public
No Inheritance
Returns a pointer to a list of Object_Set_t structures defined in the Server.H file which contains object and export information for every object registered with the server object.

`void _RegisterObject (DCEInterfaceMgr&,const boolean32)`

Privilege: Public
No Inheritance
Register the server object passing an instance of the DCEInterfaceMgr class.

5.2.2.19 Class EcClAction

Synopsis:

Parent Class: EcShAction
Parent Class: EcShActionBase
Is Not A Distributed Object
Is Associated With:
EcClSubscription (Aggregation)

Description:

A client interface object that represents the components of the action to be performed when a subscription is triggered. The possibilities are that the client will receive a notification (including all parameters that are returned by the object that triggers the subscription and an optional piece of client-specified text) and/or a request that will be executed. The client is required to specify an action for each subscription.

Attributes:

`myRequest`

Privilege: Private
Data Type: EcClRequest
Default Value: NOT IDENTIFIED
No Inheritance
The request that is currently associated with this action.

`mytext`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcShAction

`myNotify`

Privilege: Protected Attribute
Data Type: RWBoolean
Default Value: NOT IDENTIFIED
Inherited From: EcShActionBase
Whether to notify on firing

`myRequestFlag`

Privilege: Protected Attribute
Data Type: RWBoolean
Default Value: NOT IDENTIFIED
Inherited From: EcShActionBase
Whether request has been set

`myText`

Privilege: Protected Attribute
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcShActionBase
Notification text

Operations:

`EcTVoid ClearRequestB ()`

Privilege: Public
No Inheritance
Used to clear any request that has been set for the action.

`void EcCIAction ()`

Privilege: Public
No Inheritance
Default constructor

`void EcClActionB (RWCString &text, EcClRequest * = NULL)`

Privilege: Public
No Inheritance
Used to construct an action from a piece of text and, RELEASE B: optionally, a request. The notification flag is set.

`void EcClActionB (EcClRequest &, RWBoolean = FALSE, RWCString * = NULL)`

Privilege: Public
No Inheritance
Used to construct an action from a request and, optionally, a value for the notification flag and a piece of text.

`const EcClRequest& GetRequestB ()`

Privilege: Public
No Inheritance
Returns the request currently set for the action.

`void SetRequestB (EcClRequest &)`

Privilege: Public
No Inheritance
Sets the request to be executed when the subscription fires.

`void ~EcClAction ()`

Privilege: Public
No Inheritance
Destructor

`void EcShSubAction (RWCString* text)`

Privilege: Public
Inherited From: EcShAction
normal constructor

`void EcShSubAction ()`

Privilege: Protected Operation
Inherited From: EcShAction
hidden default cinstructor

`RWCString& GetText ()`

Privilege: Public
Inherited From: EcShAction
obtain notification text

`RWBoolean Notification ()`

Privilege: Public
Inherited From: EcShAction
obtain current value of notify flag

`void SetText (RWCString)`

Privilege: Public
Inherited From: EcShAction
set notification text

`void ~EcShSubAction ()`

Privilege: Public
Inherited From: EcShAction
destructor

`EcTVoid ClearRequest ()`

Privilege: Public
Inherited From: EcShActionBase
Clear any request that has been set

`void EcShActionBase (RWBoolean, RWCString)`

Privilege: Public
Inherited From: EcShActionBase

`RWBoolean GetNotify ()`

Privilege: Public
Inherited From: EcShActionBase
Obtains current value of notify flag

`RWCString GetText ()`

Privilege: Public
Inherited From: EcShActionBase
Obtains notification text

`RWBoolean HasRequest ()`

Privilege: Public
Inherited From: EcShActionBase
Find out if any request has been set

`EcTVoid SetNotify (RWBoolean)`

Privilege: Public
Inherited From: EcShActionBase

`EcTVoid SetText (RWCString)`

Privilege: Public
Inherited From: EcShActionBase

`void ~EcShActionBase ()`

Privilege: Public
Inherited From: EcShActionBase
Destructor

5.2.2.20 Class EcClSubscription

Synopsis:

Parent Class: EcShSubscription
Is Not A Distributed Object
Is Associated With:
EcClSubscriptionCollector (Aggregation)

Description:

This class is the client side subscription which can either be created from advertisements or from exisiting subscriptions from the server side (through a stream.)

Attributes:

`myAction`

Privilege: Protected Attribute
Data Type: EcClAction
Default Value: NOT IDENTIFIED
No Inheritance
The action to be performed when the subscription fires.

`myCollector`

Privilege: Protected Attribute
Data Type: EcClSubscriptionCollector&
Default Value: NOT IDENTIFIED
No Inheritance

A pointer to the collector that this reference is a member of. If this pointer is null, then this reference is a member of one of the collectors in the static collector vector.

`myDescription`

Privilege: Protected Attribute
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
String which contains service of the subscription.

`myDurationType`

Privilege: Protected Attribute
Data Type: enum EcClSubscriptionType
Default Value: {ONCE, OUTSTANDING}
No Inheritance
Time duration of subscriptions (i.e., can be done one time or forever (outstanding).

`myExpirationDate`

Privilege: Protected Attribute
Data Type: RWDate
Default Value: NOT IDENTIFIED
No Inheritance
Identifies when this subscription will expire and be removed from the system. The value may be "never" (i.e. the subscription is permanent)

`mySubmittedFlag`

Privilege: Protected Attribute
Data Type: RWBoolean
Default Value: RWTrue
No Inheritance
Flag which shows whether the subscription has been submitted or not.

`myUserInfo`

Privilege: Protected Attribute
Data Type: EcClClient
Default Value: NOT IDENTIFIED
No Inheritance
Client information, provided by client software.

`ourCollectorvector`

Privilege: Protected Attribute
Data Type: EcClSubscriptionCollectionVector
Default Value: NOT IDENTIFIED
No Inheritance
Static vector of pointers to DsClSubscriptionCollector objects, one per dataserver.

`myEventID`

Privilege: Protected Attribute
Data Type: EcTSbEventID
Default Value: NOT IDENTIFIED
Inherited From: EcShSubscription
event subscribed to

`myExpirationdate`

Privilege: Protected Attribute
Data Type: RWDate
Default Value: NOT IDENTIFIED
Inherited From: EcShSubscription
date subscription runs out

`myStartDate`

Privilege: Protected Attribute
Data Type: RWDate
Default Value: NOT IDENTIFIED
Inherited From: EcShSubscription
date subsciption was made

`myStatus`

Privilege: Protected Attribute
Data Type: EcUtStatus
Default Value: NOT IDENTIFIED
Inherited From: EcShSubscription
Status of this object

`myUserInfo`

Privilege: Protected Attribute
Data Type: MsAcUserProfile
Default Value: NOT IDENTIFIED
Inherited From: EcShSubscription
subscriber information

Operations:

`EcTVoid ClearSubmittedFlag ()`

Privilege: Protected Operation
No Inheritance
Flag as not submitted

`void EcClSubscription (userinfo, Advertisement&, EcClSubscriptionCollector&)`

Privilege: Public
No Inheritance
Constructor for client software (therefore, public) which gets attribute information from advertisements, such as service provider. If no collector has been provided, it goes and finds one, based on the static nature of the collector.

`void EcClSubscription (submittedflag, EcClSubscriptionCollector&, Stream)`

Privilege: Public
No Inheritance
Constructor for already existing collector which gets already existing subscriptions from the sever side through a stream.

`void GetAction (EcClAction)`

Privilege: Public
No Inheritance
Means of accessing myAction attribute (object), which will be communicated to the server what this subscription should do when it fires.

`EcClSubscriptionCollector* GetCollector ()`

Privilege: Protected Operation
No Inheritance
Get Collector object

`RWCString GetDescription ()`

Privilege: Public
No Inheritance
Returns description, containing the service, as a RogueWave string.

`EcEClSubscriptionType GetDurationtype ()`

Privilege: Public
No Inheritance
Accesses myDurationType attribute as to whether subscriptions are done one time or forever (outstanding).

`RWDate GetExpirationdate ()`

Privilege: Public
No Inheritance
Public access to myExpirationDate attribute, which provides the expiration date of the subscription.

`RWBoolean GetSubmittedflag ()`

Privilege: Public
No Inheritance
Public access to flag as to whether a subscription has been submitted.

`void GetUserinfo (GLClient&)`

Privilege: Public
No Inheritance
Public access to user information which can be put into the DsSrClient object.

`RWBoolean IsSubmitted ()`

Privilege: Protected Operation
No Inheritance
Indicate current state

`void SetAction (EcClAction&)`

Privilege: Public
No Inheritance
Sets the myAction attribute for this particular subscription as determined by the client software.

`void SetDescription (RWCString)`

Privilege: Public
No Inheritance
Allows the client software to fill in the Description attribute with service information.

`void SetDurationType (EcEClSubscriptionType)`

Privilege: Public
No Inheritance
Sets the attribute which determines the existence type of the subscription.

`void SetExpirationDate (RWDate)`

Privilege: Public
No Inheritance
Sets the expiration date of the subscription itself.

`void SetSubmittedFlag (RWBoolean)`

Privilege: Protected Operation
No Inheritance
Sets the flag which indicates whether or not the DsClSubscription has actually been submitted to the dataserver (i.e. the client software is finished with filling in the information, and has invoked the Submit method).

`GlStatus& Submit ()`

Privilege: Public
No Inheritance
Submits subscription to the subscription collector.

`EcUtStatus Update ()`

Privilege: Public
No Inheritance
Replace this subscription in the system

`GlStatus& Withdraw ()`

Privilege: Public
No Inheritance
Deletes a subscription from the subscription collector.

`void ~EcClSubscription ()`

Privilege: Public
No Inheritance
The DsClSubscription's destructor.

`void EcShSubscription (MsAcUserProfile)`

Privilege: Public
Inherited From: EcShSubscription
normal constructor

`void EcShSubscription (const)`

Privilege: Public
Inherited From: EcShSubscription
copy constructor

`void EcShSubscription ()`

Privilege: Public
Inherited From: EcShSubscription
default constructor defined here to prevent anyone from using it

`EcTSbEventID& GetEventID ()`

Privilege: Public
Inherited From: EcShSubscription
get value of myUserInfo

`RWDate GetExpirationDate ()`

Privilege: Public
Inherited From: EcShSubscription
get value of myUserInfo

`RWDate GetStartDate ()`

Privilege: Public
Inherited From: EcShSubscription
get value of myUserInfo

`MsAcUserProfile& GetUserInfo ()`

Privilege: Public
Inherited From: EcShSubscription
get value of myUserInfo

`EcUtStatus Getstatus ()`

Privilege: Public
Inherited From: EcShSubscription
get status of this instance

`void SetEventID (EcTSbEventID&)`

Privilege: Public
Inherited From: EcShSubscription
set myUserInfo attribute to provided info

`void SetExpirationDate (RWDate expirationDate)`

Privilege: Public
Inherited From: EcShSubscription
set myUserInfo attribute to provided info

`void SetStartDate (RWDate startDate)`

Privilege: Public
Inherited From: EcShSubscription
set myUserInfo attribute to provided info

`void SetUserInfo (MsAcUserProfile&)`

Privilege: Public
Inherited From: EcShSubscription
set myUserInfoattribute to

`void saveOn (ostream&)`

Privilege: Public
Inherited From: EcShSubscription
Debugging aid Input/Output. send msgs to

`void ~EcShSubscription ()`

Privilege: Public
Inherited From: EcShSubscription
destructor

5.2.2.21 Class EcClSubscriptionCollector

Synopsis:

Parent Class: EcClGenConnector
Distributed Object
Is Associated With:
EcClCollectorVector (Aggregation)

Description:

This public, distributed class is a specialization of the Collector class which handles DsClSubscriptions. This class provides, in addition to the normal vector operations, the ability to create a list of all subscriptions for a given user or advertisement, and a means of submitting and cancelling subscriptions. There are no attributes for this object.

Attributes:

`myStatus`

Privilege: Private
Data Type: GlStatus
Default Value: NOT IDENTIFIED
No Inheritance
This attribute allows the object to maintain information on current status.

Operations:

`const GlStatus & BuildList (Advertisement&)`

Privilege: Public
No Inheritance
This operation creates a list of all subscriptions for a given event.

`const GlStatus & BuildList (MSS_UserProfile &)`

Privilege: Public
No Inheritance
This operation creates a list of all subscriptions for a given user.

`const GlStatus & BuildListB ()`

Privilege: Public
No Inheritance
This operation allows ops/admin staff to get a list of all subscriptions in the system.

`CancelSubscription (EcClSubscription&)`

Privilege: Protection Not Identified
No Inheritance

`EcClSubscription* CreateSubscription (RWBoolean SubmittedFlag, istream &Stream, EcClSubscriptionCollector *me)`

Privilege: Private
No Inheritance
This is a private service used to build the set of subscriptions contained by the SubscriptionCollector.

`EcUtStatus CreateSubscription (GlParameterList&)`

Privilege: Protected Operation
No Inheritance
This function makes a new CsClSubscription with the provided data and adds it to the CsClSubscriptionCollector.

`void EcClSubscriptionCollector (GlUR &dataserver, MSS_UserProfile &)`

Privilege: Public
No Inheritance
The constructor for DsClSubscriptionCollector's. This constructor establishes a set of Subscriptions for the user based on the provided science data server and the user information.

`EcUtStatus Populate (GlParameterList&)`

Privilege: Protected Operation
No Inheritance
This method generates CsClSubscription objects for each of the sets of information in the provided parameter list.

`GlParameterList& SubmitRequest (EcClRequest&)`

Privilege: Public
No Inheritance

`EcTVoid SubmitSubscription (EcClSubscription&)`

Privilege: Public
No Inheritance
This function registers the provided subscription with the subscription server.

`EcTVoid saveOn (ostream&)`

Privilege: Private
No Inheritance
This function produces human-readable formatting of the contents of the object such that it can be used in the overloading of the << operator.

`void ~EcClSubscriptionCollector ()`

Privilege: Public
No Inheritance
The DsClSubscriptionCollector's destructor.

5.2.2.22 Class EcDcDSyncCom

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

This class is used to achieve message passing using asynchronous and deferred synchronous communications. It is designed to work with OODCE-provided DCE-Pthread class which is used to start and control execution of a thread.

Attributes:

`_addr`

Privilege: Private
Data Type: EcTPtr
Default Value: NULL
No Inheritance
This attribute points to some address that the developer will use in the overridden Invoke member function. It is a void pointer and it could be a port number, an IP, a binding, an object reference, a CDS name, etc.

`_call_in_progress`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
This attribute holds a value that identifies whether a thread is currently executing or not. This flag is used to assure that only one call is processed at a time, and that we don't have multiple Send calls happening concurrenlty. Only one thread will be running at a time.

`_data`

Privilege: Private
Data Type: EcTPtr
Default Value: NULL
No Inheritance
This attribute points to some data that the developer will use in the overridden invoke member function. It is a void pointer.

`_done`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
This attribute holds a value that identifies whether a thread has terminated or not. It gets updated when the process finished execution, after PostInvoke. '0' means the thread did not finish, '1' means it did.

`_noOfTries`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
This attribute defines the number of Send call re-tries if exceptions or errors occur during the communication.

`_policy`

Privilege: Private
Data Type: EcEDcThreadPolicy
Default Value: EcDDcFg
No Inheritance
This attribute represents the thread scheduling policy. It is an enum type. The policy types are: EcDDcFifo (first in/first out), EcDDcRr (Round Robin), EcDDcFg (Foreground), EcDDcBg (Background).

`_priority`

Privilege: Private
Data Type: EcEDcThreadPriority
Default Value: EcDDcPri_min
No Inheritance
This attribute represents the thread scheduling priority. It is an enum type. The priority types are: EcDDcPri_min, EcDDcPri_low, EcDDcPri_mid, EcDDcPri_hi, EcDDcPri_max.

`_results`
Privilege: Private
Data Type: EcTPtr
Default Value: NULL
No Inheritance
This attribute is used to store the results that were a product of the thread execution. It is a void pointer.

`_timeBetweenTries`
Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
This attribute defines how often Send call re-tries will occur in case of errors during the communication.

Operations:

`EcTInt CallInProgress (ao_status:EcUtStatus*)`
Privilege: Public
No Inheritance
This operation returns to the caller the execution status of the thread, whether is currently executing (in progress), or not.

`EcTInt Done (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns to the caller the thread termination status, an integer called '_done'. It is a flag and will be initialized to '0' in the constructor. When the thread finishes its execution, which is after PostInvoke, the '_done' flag is set to '1'. Once the results of the thread execution have been retrieved successfully, then Reset() can be used to clear the flag back to '0'.

`EcTVoid EcDcDSyncCom ()`

Privilege: Public
No Inheritance
Constructor.

`DCEPthreadResult EvalThread (a_param:DCEPthreadParam)`

Privilege: Public
No Inheritance
This is a static member function, an internal one. It makes calls to PreInvoke, Invoke, and PostInvoke. This function will be executing on a separate thread. It takes as input parameter a DCEPthreadParam argument, and OODCE type which represents a void pointer.

`EcTPtr GetAddr (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns a pointer to the address.

`EcTPtr GetData (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns a pointer to the data.

`EcTInt GetNoOfTries (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns the number of tries.

`EcEDcThreadPolicy GetPolicy (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns the thread scheduling policy attribute.

`EcEDcThreadPriority GetPriority (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns the thread priority attribute.

`EcTPtr GetResults (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns the method's results. This is applicable only for deferred synchronous. Once data was sent, a reply comes back with the results from the operation.

`EcTInt GetTimeBetweenTries (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation returns the time between tries.

`EcTInt Invoke ()`

Privilege: Public
No Inheritance
This operation is used to perform whatever operations the developer wishes. It is called automatically by the static function EvalThread, which is internal to CSS.

`EcTInt PostInvoke ()`

Privilege: Public
No Inheritance
This operation is used to perform whatever operations the developer wishes. It is called automatically by the static function EvalThread, which is internal to CSS.

`EcTInt PreInvoke ()`

Privilege: Public
No Inheritance
This operation is used to perform whatever operations the developer wishes. It is called automatically by the static function EvalThread, which is internal to CSS.

`EcUtStatus Reset ()`

Privilege: Public
No Inheritance
It is called after the thread has terminated, and after the results from the thread execution (if any) were retrieved. This call will reset the '_done' flag to zero, the '_call_in_progress' flag to zero, and it will deallocate any memory assigned to '_results'.

`EcTChar* Send`
`(a_return_uuid_flag:EcTInt,ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
When the Send operation is called, a thread gets created and its scheduling attributes are set. If the Send call fails, the call will be retried as many times as the developer defined in the '_noOfRetries' field. Once the thread began execution, the '_call_in_progress' flag is set to '1', the thread is terminated (but the object is not deleted), and the 'call_in_progress' flag is set to '0'. If the call was deferred synchronous, the method's results will be stored in '_results'.

`EcUtStatus SetAddr (a_addrP:EcTPtr)`

Privilege: Public
No Inheritance
This operation sets a pointer to some address that the developer wishes to use in the overridden Invoke member function. The address can be an IP, a port number, an object reference, a binding, a CDS name, etc.

`EcUtStatus SetCallInProgressFlag (a_call_in_progress:EcTInt)`

Privilege: Public
No Inheritance
This operation sets the '_call_in_progress' flag. This flag is used to identify whether a thread is currently executing or not.

`EcUtStatus SetData (a_dataP:EcTPtr)`

Privilege: Public
No Inheritance
This operation sets a pointer to some data that the developer wishes to use in the overridden Invoke member function.

`EcUtStatus SetDoneFlag (a_done:EcTInt)`

Privilege: Public
No Inheritance
This operation sets the '_done' flag. This flag is used to identify whether a thread has finished or not.

`EcUtStatus SetNoOfTries (a_noOfTries:EcTInt)`

Privilege: Public
No Inheritance
This operation sets the number of Send retries in case of exceptions/ communications errors at the other end. The Send call will be retried as many times as specified in '_noOfTries'.

```
EcUtStatus SetPolicy (a_policy:EcEDcThreadPolicy)
```

Privilege: Public
No Inheritance
This operation sets the thread scheduling policy attribute. The scheduling policies are: EcDDcFifo (first in/first out), EcDDcRr (Round Robin), EcDDcFg (Foreground), EcDDcBg (Background).

```
EcUtStatus SetPriority (a_priority:EcEDcThreadPriority)
```

Privilege: Public
No Inheritance
Threads are scheduled according to their scheduling priority and how the scheduling policy treats those priorities. This operation sets the thread priority attribute. The thread scheduling priorities are: EcDDcPri_min, EcDDcPri_low, EcDDcPri_mid, EcDDcPri_hi, EcDDcPri_max.

```
EcUtStatus SetResults (a_resultP:EcTPtr)
```

Privilege: Public
No Inheritance
This operation sets the result from the Invoke call.

```
EcUtStatus SetTimeBetweenTries (a_timeBetweenTries:EcTInt)
```

Privilege: Public
No Inheritance
This operation sets the time between each Send retry (in seconds). If the Send call failed because of communication errors, the Send call will be retried as often as specified in '_timeBetweenTries'.

```
EcTVoid ~EcDcDSyncCom ()
```

Privilege: Public
No Inheritance
Destructor.

5.2.2.23 Class EcDnAttribute

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Directory_Naming_Service (Aggregation)

Description:

The EcDnAttribute class will contain an attribute name, and type. It will also be referenced by the EcDnElement class. This class will provide methods to get the attribute name and type.

Attributes:

`_attribute_name`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Attribute name

`_attribute_type`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Attribute type.

Operations:

`void EcDnAttribute ()`

Privilege: Public
No Inheritance
Default Constructor

```
void EcDnAttribute (a_attribute_type:const
EcTInt,a_attribute_name:const RWCString&)
```

Privilege: Public
No Inheritance

`EcTChar* const GetAttributeName (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation gets the attribute name of the object.

`EcTInt const GetAttributeType (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This operation gets the attribute type of the object.

```
EcUtStatus SetAttributeName (a_attribute_name:const
RWCString&)
```

Privilege: Public
No Inheritance

```
EcUtStatus SetAttributeType (a_attribute_type:const EcTInt)
```

Privilege: Public
No Inheritance

```
void ~EcDnAttribute ()
```

Privilege: Public
No Inheritance
Default destructor.

5.2.2.24 Class EcDnCompositeName

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Directory_Naming_Service (Aggregation)

Description:

The EcDnCompositeName class will define a composite name, which is a nested set of contexts in a given hierarchy concatenated together to establish a Directory Service path name. This class will provide methods to concatenate contexts, list the contents of the composite name in the Directory Service, (soft links, object entries), read entry names, add elements (attribute/value list pair), read element information, and delete element.

Attributes:

```
_composite_name
```

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Nested set of contexts.

```
_leaf_id
```

Privilege: Private
Data Type: EcEDnCtxNameType
Default Value: NOT IDENTIFIED
No Inheritance
Leaf flag: 0 = A Context Name 1 = An Entry Name (leaf)

`ctxLst`

Privilege: Private
Data Type: EcDnContextList
Default Value: NOT IDENTIFIED
No Inheritance
Context object.

Operations:

`EcUtStatus AddAttribute (a_elt:const EcDnElement&,a_attr:const EcDnAttribute&, a_valueLst:const EcDnValueList, exist_attr_flag:const EcTBoolean)`

Privilege: Public
No Inheritance

`EcUtStatus AddCtx (a_ctx:const EcDnContext&)`

Privilege: Public
No Inheritance

`EcUtStatus AddElement (a_elt:const EcDnElement&)`

Privilege: Public
No Inheritance

`EcUtStatus AddValue (a_elt:const EcDnElement&, a_attr:const EcDnAttribute&, a_val:const EcDnValue&, exist_attr_flag:const EcTBoolean)`

Privilege: Public
No Inheritance
This operation adds a value to the value list.

`EcUtStatus AddValueList (a_elt:const EcDnElement&, a_attr:const EcDnAttribute&, a_valueLst:EcDnValueLst, exist_attr_flag:const EcTBoolean)`

Privilege: Public
No Inheritance

`EcUtStatus DeleteAttribute (a_elt:const EcDnElement&,a_attr:const EcDnAttribute&)`

Privilege: Public
No Inheritance

`EcUtStatus DeleteElement ()`

Privilege: Public
No Inheritance
This method is used to delete an element (attribute, value list) from the Directory Service.

```
DeleteValue (a_elt:const EcDnElement&,a_attr:const
EcDnAttribute&, a_val:const EcDnValue&)
```

Privilege: Protection Not Identified
No Inheritance

```
EcUtStatus DeleteValueList (a_elt:const
EcDnElement&,a_attr:const
EcDnAttribute&,a_valueLst:EcDnValueList)
```

Privilege: Public
No Inheritance

```
void EcDnCompositeName ()
```

Privilege: Public
No Inheritance
Default Constructor.

```
void EcDnCompositeName (a_string:const
RWCString,a_flag:const EcEDnCtxNameType)
```

Privilege: Public
No Inheritance
Constructor number two. It will be used in the case in which DCE names are stored directly into the DCE Directory Service (DNS/GDS). It takes a string which is the full Directory Service cell name. This can be used in place of using the first constructor and adding on to that composite name via the add_ctx method.

```
void EcDnCompositeName (a_ctx:const EcDnContext&)
```

Privilege: Public
No Inheritance

```
EcTChar* const GetCompositeName (ao_status:EcUtStatus*)
```

Privilege: Public
No Inheritance
This method is used to retrieve the composite name of the object.

```
EcEDnCtxNameType const GetLeafId (ao_status:EcUtStatus*)
```

Privilege: Public
No Inheritance
This method is used to retrieve the leaf id.

```
EcUtStatus ListCtx (ao_CompositeLst:EcDnCompositeNameList*)
```

Privilege: Public
No Inheritance
This method is used to list the contents of the composite name in the Directory Service (soft links, object entries). Initiates the enumeration process for the target context. It returns a handle to a ECSRWCtxListP object (RW) used to enumerate the immediate subordinates of the named entry.

```
EcUtStatus ListElement (ao_attrList:EcDnAttributeList)
```

Privilege: Public
No Inheritance
This method is used to list the contents of the composite name (entry name) in the Directory Service. Using the composite name (distinguished name) and the attribute type, this method will return a list of attributes corresponding to the entry given.

```
EcUtStatus ModifyAttribute (a_elt:const
EcDnElement&,a_old_attr_obj:const EcDnAttribute&
a_new_attr_obj:const EcDnAttribute&,a_valueLst:const
EcDnValueList)
```

Privilege: Public
No Inheritance

```
EcUtStatus ModifyValue (a_elt:const
EcDnElement&,a_attr:const
EcDnAttribute&,a_old_value_obj:const EcDnValue&,
a_new_value_obj:const EcDnValue&)
```

Privilege: Public
No Inheritance

```
EcUtStatus ReadElement (a_attr:const
EcDnAttribute&,ao_elt:EcDnElement**)
```

Privilege: Public
No Inheritance

```
EcUtStatus ReadElement (a_attribute_name:const
RWCString&,ao_elt:EcDnElement**)
```

Privilege: Public
No Inheritance

```
EcUtStatus SetCompositeName (a_comp_name:const RWCString&)
```

Privilege: Public
No Inheritance

```
EcUtStatus SetLeafId (a_leaf_id:const EcEDnCtxNameType)
```

Privilege: Public
No Inheritance
This method is used to set the leaf id.

```
void ~EcDnCompositeName ()
```

Privilege: Public
No Inheritance
Default Destructor.

5.2.2.25 Class EcDnContext

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Directory_Naming_Service (Aggregation)

Description:

The EcDnContext class defines the path/set of bindings with distinct atomic names. Every context has an associated naming convention. An EcDnContext object is passed to the EcDnCompositeName object in a structural form as an ordered sequence of components

Attributes:

```
_cell_flag
```

Privilege: Private
Data Type: EcEDnCellType
Default Value: NOT IDENTIFIED
No Inheritance
Flag:0 = global root name (/...), 1 = cell root name (/.:)

```
_context_name
```

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Context name (a partial name of the distinguished name)

`_leaf_flag`

Privilege: Private
Data Type: EcEDnCtxNameType
Default Value: NOT IDENTIFIED
No Inheritance
Flag:0 = directory path, 1 = entry path (leaf/object), 2 = Any other type.

Operations:

`void EcDnContext ()`

Privilege: Public
No Inheritance
Default constructor.

`void EcDnContext (a_flag:const EcEDnCellType)`

Privilege: Public
No Inheritance
Constructor number one. This constructor will be used to define the type of name to be used in the DCE environment (global root name or cell root name type)

`void EcDnContext (a_str:const RWCString&,a_flag:const EcEDnCtxNameType)`

Privilege: Public
No Inheritance
Constructor number two. This constructor will be used to define a context that can be used to build the composite name.

`EcEDnCellType const GetCellFlag (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This method will return the cell flag

`RWCString const GetContextName (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This method will return the context name

`EcEDnCtxNameType const GetLeafFlag (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This method will return the leaf flag

`EcUtStatus SetCellFlag (a_cell_flag:const EcEDnCellType)`

Privilege: Public
No Inheritance
This method will set the cell flag

`EcUtStatus SetContextName (a_context_name:const RWCString&)`

Privilege: Public
No Inheritance

`EcUtStatus SetLeafFlag (a_leaf_flag:const EcEDnCtxNameType)`

Privilege: Public
No Inheritance
This method will set the leaf flag

`void ~EcDnContext ()`

Privilege: Public
No Inheritance
Default Destructor

5.2.2.26 Class EcDnElement

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Directory_Naming_Service (Aggregation)

Description:

The EcDnElement class will contain an element, which is an attribute-value list pair. It will be referenced by the EcDnCompositeName class. This class will provide methods to add value(s), get value list, delete value(s), modify value(s) and get the element name.

Attributes:

`_attribute_object`

Privilege: Private
Data Type: EcDnAttribute
Default Value: NOT IDENTIFIED
No Inheritance
Attribute object.

```
_valueLst
```

Privilege: Private
Data Type: EcDnValueList
Default Value: NOT IDENTIFIED
No Inheritance
RW list of value objects

Operations:

```
EcUtStatus AddAttribute (a_comp_name:const
RWCString&,a_attr:const
EcDnAttribute&,a_valueLst:EcDnValueList,
exist_attr_flag:const EcTBoolean)
```

Privilege: Public
No Inheritance

```
EcUtStatus AddValue (a_comp_name:const
RWCString,a_attr:const EcDnAttribute&,a_val:const
EcDnValue&)
```

Privilege: Public
No Inheritance

```
EcUtStatus DeleteAttribute (a_comp_name:const
RWCString&,a_attr:const EcDnAttribute&)
```

Privilege: Public
No Inheritance

```
EcUtStatus DeleteValue (a_comp_name:const
RWCString,a_attr:const EcDnAttribute&,a_val:const
EcDnValue&)
```

Privilege: Public
No Inheritance

```
void EcDnElement ()
```

Privilege: Public
No Inheritance
Default Constructor

```
void EcDnElement (a_attr:const EcDnAttribute&,a_vlst:const
EcDnValueList)
```

Privilege: Public
No Inheritance

```
EcDnAttribute& const GetAttributeObject
(ao_status:EcUtStatus*)
```

Privilege: Public
No Inheritance
This operation reads/gets the value list of the element set by the user which will be used when adding the element into the Directory Name space.

```
EcDnValueList const GetValueList (ao_status:EcUtStatus*)
```

Privilege: Public
No Inheritance
This operation reads/gets the value list of the element set by the user which will be used by add_element() when an entry is added. This value list is in memory (a RW list type).

```
EcUtStatus SetAttributeObject (a_attribute_object:const
EcDnAttribute&)
```

Privilege: Public
No Inheritance

```
EcUtStatus SetValueList (a_ValueLst:const EcDnValueList&)
```

Privilege: Public
No Inheritance

```
void ~EcDnElement ()
```

Privilege: Public
No Inheritance
Default Destructor.

5.2.2.27 Class EcDnValue

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Directory_Naming_Service (Aggregation)

Description:

The EcDnValue class defines a value. It will also be referenced by the EcDnElement and EcDnAttribute class.

Attributes:

`_value`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Value name

Operations:

`void EcDnValue ()`

Privilege: Public
No Inheritance
Default Constructor

`void EcDnValue (a_value:const RWCString&)`

Privilege: Public
No Inheritance

`EcTChar* const GetValue (ao_status:EcUtStatus*)`

Privilege: Public
No Inheritance
This method will return the value.

`EcUtStatus SetValue (a_value:const RWCString&)`

Privilege: Public
No Inheritance

`void ~EcDnValue ()`

Privilege: Public
No Inheritance
Default destructor.

5.2.2.28 Class EcMpMsgCb

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcMpSessionList(Public) hasa

Description:

This class will handle two types of callbacks: 1. For ordinary receive messages: If an ordinary message is received, then HandleMsg is invoked; 2. For acknowledgment of messages: If an acknowledgement is received, then HandleAck is invoked. A sending session is basically a point to point session to send a message. Each sender session will have a logical name that is needed to contact the receiver. A list of sending sessions is maintained in a given application. A sender session list contains a callback object which provides virtual funtions to be called when a send is complete. This is done at the sender side. A callback object is created and will implement the acknowledgment. The virtual function HandleAck is called when a message is delivered to the destination or when the underlying mechanism failed to deliver it within the given constraints (number of tries)

Attributes:

Operations:

```
EcTVoid EcMpMsgCb ()
```

Privilege: Public
No Inheritance
Constructor.

```
EcTVoid HandleAck (msgP:EcTVoid*,msgLength:EcTInt,
msgId:EcTChar*,SenderAddr:EcTChar*,
recAddr:EcTChar*,priority:EcTInt,sourceUUID:EcTChar*,PassFa
iledSt:EcTInt)
```

Privilege: Public
No Inheritance
Callback for 'send with acknowledgment', an asynchronous type of send.

```
EcTVoid HandleMsg (msgP:EcTVoid* msgLength:EcTInt
msgId:EcTChar* senderAddr:EcTChar* priority:EcTInt)
```

Privilege: Public
No Inheritance
Callback for ordinary receive.

```
EcTVoid ~EcMpMsgCb ()
```

Privilege: Public
No Inheritance
Destructor.

5.2.2.29 Class EcMpMsgPsngCtrl

Synopsis:

Parent Class: RWFile
Is Not A Distributed Object
Is Associated With:
Class: EcMpQueue(Private) initializes
Message_Passing_Service (Aggregation)

Description:

This class is the controller object, through which any number of receiver sessions can be created. Each receiver session is associated with a unique name (so other applications can send messages to this receiver) and a unique file (optional).The file is used for persistence. When a message comes in, it is stored in the queue associated with the object and a copy of it is stored in the file associated with this object. Internally, this object creates a sender queue. All outgoing messages are kept in this queue. A number of threads are generated internally in the initialize call which periodically get messages from the outgoing queue and send them.

Attributes:

`_cacheListP`

Privilege: Private
Data Type: CacheList *
Default Value: NOT IDENTIFIED
No Inheritance
This attribute is a two dimensional array of a hash table containg a list of pairs, an application name and the corresponding proxy object it is pointing to. The pair gets removed from the list when we find out that the server is not longer listening.

`_interfaceUuid`

Privilege: Private
Data Type: EcTChar *
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents an interface uuid.

`_objUuid`

Privilege: Private
Data Type: EcTChar *
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents an object uuid.

`_theMsgPsngCtrlP`

Privilege: Private
Data Type: EcMpMsgPsngCtrl *
Default Value: NOT IDENTIFIED
No Inheritance
This attribute is a pointer to the EcMpMsgPsngCtrl object, a global one.

Operations:

`EcTVoid Cleanup ()`

Privilege: Public
No Inheritance
This operation deallocates any memory used and destroys/shutdowns the threads.

```
EcMpMsgQueueIn* CreateReceiver
(recSessionName:EcTChar*,diskFileName:EcTChar*)
```

Privilege: Public
No Inheritance
A receiver is created by means of the CreateReceiver operation. Each receiver session is associated with a unique name (so other applications can send messages to this receiver), and a unique file. The file name is used for persistence. When a message comes in, it is stored into the queue associated with the object and a copy of it is stored on the file associated with this object. CreateReceiver will create/initialize the ordinary receiver queue, and open the file for persistence purposes. It will also create an instance of the transfer server object, and will attach a pointer to the proper queue.

```
EcMpMsgQueueInCb* CreateReceiverCB
(recSessionName:EcTChar*,diskFileName:EcTChar*,cbObj:EcMpMs
gCb*)
```

Privilege: Public
No Inheritance
A special receiver can be created by means of this operation, it calls the user-defined call back function. A thread, just one, will be awakened every time a message arrives into the special reciver queue created by the call 'CreateReceiverCB'and a call back (a virtual function) will be executed. Then, it will check if there is any other message in the queue and if so, execute the call back again or else shutdown the thread until a message arrives in the queue. CreateReceiverCb will create/initialize the callback receiver queue and open its respective file for persistence purposes. It will also create an instance of the transfer server object, and it will attach a pointer to the proper queue.

`EcTVoid EcMpMsgPsngCtrl (filename:EcTChar*,appName:EcTChar*)`

Privilege: Public
No Inheritance
This operation is the constructor. It takes two parameters: a filename and an application name. The file name will be used to name: . the five outgoing queues, one for each priority type . the five outgoing disk

storage files - persistence . the incoming queue . the incoming disk storage file - persistence The application name will be a full CDS entry path name. It will represent the relative name in the CDS where the server will store its binding information. The caller's application store its binding in this application name. The CDS directory path should be created by default in the developer's cell with public write permissions. The caller needs to be authenticated in order to write the bindings into the CDS. At construction time the following actions take place: . Initialize the Outgoing/Incoming queues. . Open the in/out disk files with read/write permissions. . Set the thread scheduling attributes for outogoing queues . Using appl. name,checks whether a thread is listening.

`EcTInt Initialize ()`

Privilege: Public
No Inheritance
This operation starts listening if it is not already.

```
EcMpQueueOut* PutMessage
(msgP:EcTVoid*,msgLength:EcTInt,msgId:EcTChar*,recAddr:EcTC
har*, senderAddr:EcTChar*,priority:EcTInt)
```

Privilege: Public
No Inheritance
This operation calls the EcMpQueueOut method Send to send a message.

`EcTVoid ~EcMpMsgPsngCtrl ()`

Privilege: Public
No Inheritance
This operation is the default destructor.

5.2.2.30 Class EcMpQueueCbIn

Synopsis:

Parent Class: EcMpQueue
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcMpQueue

Description:

This queue will contain one thread which will execute a callback every time a message is received. The callback will be a virtual function call. This class defines a double linked list queue. It inherits from the Rogue Wave Library file, RWTPtrDlist.

Attributes:

`_CBP`

Privilege: Private
Data Type: EcMpMsgCb*
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents a callback pointer.

`_QueueItems`

Privilege: Private
Data Type: EcTMpQItem
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents a queue item type.

`_TransferSrvP`

Privilege: Private
Data Type: EcMpTransferSrv*
Default Value: NOT IDENTIFIED
Inherited From: EcMpQueue
This attribute represents a pointer to the EcMpTransferSrv object.

Operations:

`EcTVoid EcMpQueueCbIn ()`

Privilege: Public
No Inheritance
This operation is the constructor.

`EcTVoid InitQ ()`

Privilege: Public
No Inheritance
This operation initializes the queue. Calls RWTPtrDlist<inCbList> to construct an empty list. It does also any other initialization steps that apply to the class EcMpMsgQueueCbIn.

`EcTInt RemoveMessage ()`

Privilege: Public
No Inheritance
This operation removes all the flagged items. This function checks for those messages with a deletion flag on, and calls the RW remove call to delete the message from the queue. This function will be called periodically.

`EcTInt SendMessage (msgP:EcTVoid*,msgLength:EcTInt, msgId:EcTChar*, senderAddr:EcTChar*, priority:EcTInt)`

Privilege: Public
No Inheritance
Once a message arrives in the queue, this operation will wake up a thread (there will be only one thread) and it will invoke the virtual callback function 'HandleMsg' to let the receiver know that a message just arrived in the queue. Next, it will search to check if another message arrived in the queue, and if so it will invoke another callback. If there are no messages in the queue, the thread will be shutdown until a message arrives in which case it will be awakened again.

`EcTVoid SetCallback (CbP:EcMpMsgCb*)`

Privilege: Public
No Inheritance
This operation sets a pointer to the callback object.

`EcTVoid ~EcMpQueueCbIn ()`

Privilege: Public
No Inheritance
This operation is the default destructor.

`EcTVoid EcMpQueue ()`

Privilege: Public
Inherited From: EcMpQueue
This operation is the constructor.

`EcTInt GetMessage ()`

Privilege: Public
Inherited From: EcMpQueue
This operation performs a single read operation and returns a message from the Incoming queue or a return status. This call will visit the next node in the queue list. If an attempt was made to read beyond the tail, it will return CsCRemoveQueueFailed. If the queue is empty, it will return CsCQueueIsEmpty. If the read operation was successful, it will return CsCSuccess.

`EcTInt GetMessageWait ()`

Privilege: Public
Inherited From: EcMpQueue
This operation performs a single read operation; however, this call waits until a message is read from the queue. Optionally a time to wait may be provided in the wait call. It actually removes the head node from the queue. Returns either CsCSuccess, CsCRemoveQueueFailed, or CsCTimeout.

`EcTVoid InitQ ()`

Privilege: Public
Inherited From: EcMpQueue
This operation initializes the queue. Calls RWTPtrDlist<List> to construct an empty list. It does also any other initialization steps that apply to the class EcMpMsgQueue.

`EcTInt RemoveMessage ()`

Privilege: Public
Inherited From: EcMpQueue
This operation removes all the flagged items. This function checks for those messages with a deletion flag on, and calls the RW remove call to delete the message from the queue. This function will be called periodically.

`EcTInt SendMessage ()`

Privilege: Public
Inherited From: EcMpQueue
This operation sends a message.

`EcTInt SetTransferServerPtr (transferSrvP:EcMpTransferSrv*)`

Privilege: Public
Inherited From: EcMpQueue
This operation sets a pointer to the transfer server class.

`EcTVoid ~EcMpQueue ()`

Privilege: Public
Inherited From: EcMpQueue
This operation is the default destructor.

5.2.2.31 Class EcMpQueueIn

Synopsis:

Parent Class: EcMpQueue
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcMpQueue

Description:

This class will be used to queue the messages once they are received. It will provide a Read Wait call and a Read Non-Wait call. The Read Wait call performs a single read operation; however, this call waits until a message is read from the queue. Optionally a time to wait may be provided in the wait call or a default time will be used. The Read Non-

Wait performs a single read operation and returns a message from the Incoming Queue (or Null if there are no messages in the queue) and a return status. This class defines a double linked list queue. It inherits from the Rogue Wave Library file, RWTPtrDlist.

Attributes:

`_QueueItems`

Privilege: Private
Data Type: EcTMpQtem
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents a queue item type.

`_TransferSrvP`

Privilege: Private
Data Type: EcMpTransferSrv*
Default Value: NOT IDENTIFIED
Inherited From: EcMpQueue
This attribute represents a pointer to the EcMpTransferSrv object.

Operations:

`EcTVoid EcMpQueueIn ()`

Privilege: Public
No Inheritance
Default constructor.

`EcTInt GetMessage (msgP:EcTVoid*)`

Privilege: Public
No Inheritance
This operation performs a single read operation and returns a message from the incoming queue or a return status. This call will visit the next node in the queue list. If an attempt was made to read beyond the tail, it will return CsCRemoveQueueFailed. If the queue is empty, it will return CsCQueueIsEmpty. If the read operation was successful, it will return CsCSuccess.

`EcTInt GetMessageWait (msgP:EcTVoid*, timeout:EcTInt)`

Privilege: Public
No Inheritance
This operation performs a single read operation; however, this call waits until a message is read from the queue. Optionally a time to wait may be provided in the wait call. It actually removes the head node from the queue and returns either CsCSuccess, CsCRemoveQueueFailed, or CsCTimeout.

`EcTVoid* InitQ ()`

Privilege: Public
No Inheritance
This operation initializes the queue. It calls RWTPtrDlist<inList> to construct an empty list. It does also any other initialization steps that apply to the class EcMpMsgQueueIn.

`EcTInt RemoveMessage ()`

Privilege: Public
No Inheritance
This operation removes all the flagged items. This function checks for those messages with a deletion flag on, and calls the RW remove call to delete the message from the queue. This function will be called periodically.

`EcTVoid ~EcMpQueueIn ()`

Privilege: Public
No Inheritance
Default destructor.

`EcTVoid EcMpQueue ()`

Privilege: Public
Inherited From: EcMpQueue
This operation is the constructor.

`EcTInt GetMessage ()`

Privilege: Public
Inherited From: EcMpQueue
This operation performs a single read operation and returns a message from the Incoming queue or a return status. This call will visit the next node in the queue list. If an attempt was made to read beyond the tail, it will return CsCRemoveQueueFailed. If the queue is empty, it will return CsCQueueIsEmpty. If the read operation was successful, it will return CsCSuccess.

`EcTInt GetMessageWait ()`

Privilege: Public
Inherited From: EcMpQueue
This operation performs a single read operation; however, this call waits until a message is read from the queue. Optionally a time to wait may be provided in the wait call. It actually removes the head node from the queue.   Returns either CsCSuccess, CsCRemoveQueueFailed, or CsCTimeout.

`EcTVoid InitQ ()`

Privilege: Public
Inherited From: EcMpQueue
This operation initializes the queue. Calls RWTPtrDlist<List> to construct an empty list. It does also any other initialization steps that apply to the class EcMpMsgQueue.

`EcTInt RemoveMessage ()`

Privilege: Public
Inherited From: EcMpQueue
This operation removes all the flagged items. This function checks for those messages with a deletion flag on, and calls the RW remove call to delete the message from the queue. This function will be called periodically.

`EcTInt SendMessage ()`

Privilege: Public
Inherited From: EcMpQueue
This operation sends a message.

`EcTInt SetTransferServerPtr (transferSrvP:EcMpTransferSrv*)`

Privilege: Public
Inherited From: EcMpQueue
This operation sets a pointer to the transfer server class.

`EcTVoid ~EcMpQueue ()`

Privilege: Public
Inherited From: EcMpQueue
This operation is the default destructor.

5.2.2.32 Class EcMpSessionList

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcMpMsgCb(Public) hasa
Class: EcMhPendingMsg(Private) send
Message_Passing_Service (Aggregation)

Description:

This is a container class whose element type is a logical name and will inherit from the RWTPtrSlist class. A session list contains a callback object which provides virtual functions to be called when a send is complete. This is done at the sender side.

Attributes:

`_NoOfTries`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Number of tries in case of communication errors.

`_timeBetweenTries`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Number of seconds between tries.

Operations:

`EcTInt Delete (LogicalName:EcTChar*)`

Privilege: Public
No Inheritance
Deletes a receiver, a logical name from the list.

`EcTVoid EcMpSessionList (CB:EcMpMsgCb*)`

Privilege: Public
No Inheritance
Constructor.

`EcTInt Join (logicalName:EcTChar*)`

Privilege: Public
No Inheritance
Used to insert a logical name into the list.

```
EcTInt Send
(syncAsyncFlag:EcTInt,msgP:EcTVoid*,msgLength:EcTInt,
recAddr:EcTChar*,senderAddr:EcTChar*,priority:EcTInt,msgUui
dFlag:EcTInt, msgId:EcTChar*)
```

Privilege: Public
No Inheritance
Sends a message to a particular session. The message can be sent expecting an acknowledgment or not expecting an acknowledgment. Internally if the flag is synchronous, the remote function, transmit, will be called and the message sent at once. Control will return when the Send operation completed. If the flag is asynchronous, the message will be placed in the outgoing queue and it will be sent from there.

`EcTVoid SetTries (i_tries:EcTInt i_timeBetweenTries:EcTInt)`

Privilege: Public
No Inheritance
Set the number of tries if a message fails to be sent and needs to be retried, and the time between the tries.

`EcTVoid ~EcMpSessionList ()`

Privilege: Public
No Inheritance
Destructor.

5.2.2.33 Class EcPfManagedServer

Synopsis:

Parent Class: EcPfGenServer
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcPfGenServer

Description:

This class is the framework class for Managed Server Processes. This class defines the method Process Event which handles the events generated by the Managed Server Processes. This class is also connected to the MSS EcAgManager class as required by the MSS desing. The Managed Server class will povide methods to inform the EcAgManager to start and stop monitoring, to inform the EcAgManager the number of shutdown seconds required for the application, program, and process, to register metrics with the EcAgManager. The Managed Server class also receives requests from the EcAgManager class to suspend, resume, and shutdown. The method, PfShutdownMyself, is provided by the Managed Server class to the application may request a shutdown of itself.

Attributes:

`myMSSMgrPtr`

Privilege: Private
Data Type: EcAgManager*
Default Value: NOT IDENTIFIED
No Inheritance
Pointer to the EcAgManger object. Used to call methods contained in the EcAgManager class.

`GroupName`

Privilege: Private
Data Type: RWCString
Default Value: NULL
Inherited From: EcPfGenServer
Needed to decide the group name in CDS

`MyPolicy`

Privilege: Private
Data Type: HostPolicy
Default Value: 1
Inherited From: EcPfGenServer
Decides the host policy; 0 for unique host policy, 1 for multiple host policy.

`NbrOfFTPThr`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcPfGenServer
Indicates the number of FTP threads in the application; The FTP initialization is done only for values greater then 0.

`ObjectCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Keeps track of the number of objects in the link list

`ObjectLinkListPtr`

Privilege: Private
Data Type: Pointer to the ObjectLinkList class
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Pointer to the linked list of objects to be registered or unregistered.

`PrincipalName`

Privilege: Private
Data Type: RWCString
Default Value: NULL
Inherited From: EcPfGenServer
Needed in message passing and security, user name

`ProfileName`

Privilege: Private
Data Type: RWCString
Default Value: NULL
Inherited From: EcPfGenServer
Needed to decide the profile name in CDS

`ProtocolPolicy`

Privilege: Private
Data Type: RWCString
Default Value: NULL
Inherited From: EcPfGenServer
Needed to decide the protocol policy of CDS

`SRFflag`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Flag indicating whether Server Request Framework is needed

`ServerStatus`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcPfGenServer
There is to indication the status of server objects. There are three different status: 1) Initial - no user objects registered with the server, 2) suspendable and 3) resumable

`Server_state_mutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: unlocked
Inherited From: EcPfGenServer
Lock attribute used to activate a self unlocking mutex that forces register object, unregister object, suspend, resume, to be called sequentially.

`aclDBName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Filename for ACL DB

`diskFileName`

Privilege: Private
Data Type: RWCString
Default Value: NULL
Inherited From: EcPfGenServer
Needed as a parameter in message passing

`keyfile`

Privilege: Private
Data Type: RWCString
Default Value: NULL
Inherited From: EcPfGenServer
Needed as a parameter in message passing and security

`messpassflag`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Flag that is put to 1 by the user in order to explicitly require that message passing be activated.

`rwHashTable`

Privilege: Private
Data Type: RWHashDictionary
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Contains FTP thread index and associated callback functions

`serverFTPptr`

Privilege: Private
Data Type: CsFtFTPSchedObj*
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
Pointer to FTP object

`serverShortName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenServer
server name

`myAppID`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Application ID.

`myAppName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Name of the application

`myConfigFileName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Full configuration file name path. It must be provided in the command line.

`myConfigFileP`

Privilege: Private
Data Type: EcPfConfigFile*
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
A pointer to the configuration file.

`myDeltaTime`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Delta time for simulation purposes

`myExecName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Executable name of the program

`myMajorVersion`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcPfGenProcess
Major version of the application

`myMinorVersion`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcPfGenProcess
Minor version of the application.

`myMode`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Provides the mode the user is curently in (i.e. test, production, training, etc...). It must be privided from the command line.

`myPID`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Process ID

`myPath`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
CDS entry path.

`myProgramID`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Program ID.

`mySite`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: EcPfGenProcess
Site name where the process is running.

Operations:

```
void EcPfManagedServer
(a_argc:ECTInt,a_argv:EcTChar**,status:EcUtStatus)
```

Privilege: Public
No Inheritance
Constructor

`EcUtStatus PfExecShutdown (a_level:EcAgMgmtLevel)`

Privilege: Public
No Inheritance
Calls theServer Shutdown() and PfShutdown method with the management level(application, program, process).

`EcTInt PfGetShutdownSeconds (a_level:EcTAgMgtmLevel)`

Privilege: Public
No Inheritance
Needs to be overriden by application, the number of seconds requires to shutdown the application

`EcUtStatus PfInit ()`

Privilege: Public
No Inheritance
Calls the PfGenServerInit(inherited) method, instantiates the EcAgManager Object and registers it with the GSO

```
EcUtStatus PfProcessEvent
(a_event:EcAgEvent*,a_log_type:EcTAgLogType)
```

Privilege: Public
No Inheritance
Calls the ProcessEvent method of the EcAgManager object

```
EcUtStatus PfRegisterMetric (a_level
EcAgMgmtLevel,a_perfmetric:EcAgPerfMetric*)
```

Privilege: Public
No Inheritance
Calls the RegisterMetric method of the EcAgManger object.

```
EcUtStatus PfRegisterMetric (a_level
EcAgMgmtLevel,a_faultmetric:EcAgFaultMetric*)
```

Privilege: Public
No Inheritance
Calls the RegisterMetric method of the EcAgManager object.

```
EcUtStatus PfRegisterMetric
(a_level:EcAgMgmtLevel,a_configmetric:EcAgConfigMetric*)
```

Privilege: Public
No Inheritance
Calls the RegisterMetric method of the EcAgManager object.

```
EcUtStatus PfShutdown
(shutdownlevel:EcTAgMgmtLevel,EcTInt,EcTInt)
```

Privilege: Public
No Inheritance
Needs to be overriden by application to perform specific shutdown appropriate for application

```
EcUtStatus PfShutdownMyself
(a_level:EcAgMgmtLevel,a_event:EcAgEvent)
```

Privilege: Public
No Inheritance
Calls StopMonitoring. Calls PfShutdown method with the management level(application,program,process). Calls the DCEServer Shutdown method.

```
EcUtStatus PfStart ()
```

Privilege: Public
No Inheritance
Calls PfStartMonitoring and DCEServer listen methods

`EcUtStatus PfStartMonitoring ()`

Privilege: Private
No Inheritance
Calls StartMonitoring() method of the EcAgManager Object

`EcUtStatus PfStopMonitoring (EcTInt)`

Privilege: Private
No Inheritance
Calls StopMonitoring() method of the EcAgManager Object

`void ~EcPfManagedServer ()`

Privilege: Public
No Inheritance
Destructor. Deletes the EcAgManager object.

`void EcPfGenServer (EcTInt, EcTChar**, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
EcPfGenServer Constructor

`EcUtStatus PfAddCb (index:EcTInt,cb:EcMpMsgCb*)`

Privilege: Public
Inherited From: EcPfGenServer
This will calls the respective message passing method. (For details see description of EcMpMsgPsngCtrl class)

`EcUtStatus PfAddIndexAndCallback (EcTInt,EcTVoid)`

Privilege: Public
Inherited From: EcPfGenServer
This method is called by the application programmer any time they make a FTP call. Its purpose is to keep the callback together with its index in a HashTable, so that in case of failure, given the index, the address of the callback can be traced.

`RWCString PfCreateFileAndModeName (RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
This method is used to create filenames with the extension _mode, so that be explicitly forced that when applications are run in different modes, the files used for Security, Message Passing, etc, be different.

`EcMpMsgQueueIn* PfCreateReceiver (RWCString&recSessName,RWCString&diskFileNameEcMpMsgCb*cbObj,EcU tStatus&status)`

Privilege: Public
Inherited From: EcPfGenServer
Method for creating a receiver object for message passing

`EcMpMsgQueueCbIn* PfCreateReceiverCb (RWCString,RWCString,EcTInt,DCEUuid,RWCString,EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Method for creating a receiver object for message passing

`EcMpSessionList* PfCreateSessionList (EcTInt, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Creates session list for message passing

`EcUtStatus PfGenServerInit ()`

Privilege: Public
Inherited From: EcPfGenServer
Starts the main attributes of the server obtained in configuration file and command line

`EcTInt PfGetCallbackAddress (EcTInt, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
This method is called by FTP service in case of failure. Given the index, it will trace the address of the callback.

`EcUtStatus PfRegisterList ()`

Privilege: Private
Inherited From: EcPfGenServer
This method is called by PfResume for re-registering objects with the server

`EcUtStatus PfRegisterObject (DCEObj&, dceFlag:boolean)`

Privilege: Public
Inherited From: EcPfGenServer
Method for registering objects with the server and inserting them in a private list

`EcUtStatus PfRemoveObjectOfList (DCEObj&)`

Privilege: Private
Inherited From: EcPfGenServer
This method is called by PfUnregisterObject for removing an objects from the linked list

`EcUtStatus PfResume ()`

Privilege: Public
Inherited From: EcPfGenServer
Method for re-registering the objects that reside in the linked list

`EcUtStatus PfSetAttrFromArgv (argc:int,  argv:char**)`

Privilege: Private
Inherited From: EcPfGenServer
This method is called by the constructor to check for any possible attributes that need to be set. It will overwrite the attribute values already read in the configuration file.

`EcTVoid PfSetAttrFromConfigFile (EcUtStatus*)`

Privilege: Private
Inherited From: EcPfGenServer
Read the configuration file and sitting class attribuites accordingly

`EcUtStatus PfSuspend ()`

Privilege: Public
Inherited From: EcPfGenServer
Method for unregistering all objects that are inserted in the linked list

`EcUtStatus PfUnregisterList ()`

Privilege: Private
Inherited From: EcPfGenServer
This method is called by PfSuspend for unregistering objects from the server

`EcUtStatus PfUnregisterObject (DCEObj&, dceFlag:boolean)`

Privilege: Public
Inherited From: EcPfGenServer
Method for unregistering a given object with the server and removing it from the private list

`EcUtStatus PfUnregisterObject (uuid_t*, dceFlag:boolean)`

Privilege: Public
Inherited From: EcPfGenServer
Method for unregistering a given object with the server and removing it from the private list

`RWCString Pfget_GroupName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns GroupName value

`HostPolicy Pfget_HostPolicy (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Return HostPolicy value

`EctInt Pfget_NbrOfFTPThr (Status: EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns NbrOfFTPThr value

`RWCString Pfget_PrincipalName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns PrincipalName value

`RWCString Pfget_ProfileName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns the ProfileName value

`RWCString Pfget_ProtocolPolicy (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns the ProtocolPolicy value

`RWCString Pfget_SRFflag (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Return Server Request Framework flag

`EctInt Pfget_ServerStatus (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns server status _ServerStatus

`RWCString Pfget_aclDBName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns file name for ACL database

`RWCString Pfget_diskFile (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns diskFile value

`RWCString Pfget_keyFile (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns keyfile value

`CsFtFTPSchedObj* Pfget_serverFTPptr (EcUtstatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Returns pointer to FTP object

`void Pfset_GroupName (const RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set group name

`EctVoid Pfset_HostPolicy (HostPolicy, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set HostPolicy value

`void Pfset_NbrOfFTPThr (const EcTInt, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set NbrOfFTPThr value

`void Pfset_PrincipalName (const RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set PrincipalName value

`void Pfset_ProfileName (const RWCString,EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
set profile name value

`void Pfset_ProtocolPolicy (const RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set ProtocolPolicy value

`EcTVoid Pfset_aclDBName (RWCString, EcUtStatus)`

Privilege: Public
Inherited From: EcPfGenServer
Sets the ACL DB file name attribute

`void Pfset_diskFile (const RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set diskFile value

`void Pfset_keyfile (const RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenServer
Set keytab file location

`EcTVoid Shutdown ()`

Privilege: Protected Operation
Inherited From: EcPfGenServer
This is theServer Shutdown method

`EcUtStatus StartDceServerProc ()`

Privilege: Private
Inherited From: EcPfGenServer
This method is called privately by the PfGenStart to initialize common server activities such as SetName, SetProtocols, etc. It will only perform those activities for which the respective attributes will be set either in the configuration file or in the command line.

`EcUtStatus StartFTPProc ()`

Privilege: Private
Inherited From: EcPfGenServer
This method is called privately by the PfGenStart to initialize the FTP process when the application sets the number of FTP threads to a value greater than 0. The maximum number of FTP threads allowed is 30.

`EcUtStatus StartMessPassProc ()`

Privilege: Private
Inherited From: EcPfGenServer
This method is called privately by the PfGenStart to initialize the message passing process.

`EcUtStatus StartSecurityProc ()`

Privilege: Private
Inherited From: EcPfGenServer
This method is called by PfGenServerInit to initialize the Security process.

`void ~EcPfGenServer ()`

Privilege: Public
Inherited From: EcPfGenServer
Destructor

`void EcPfGenProcess`
`(argc:EcTInt,argv:EcTChar**,status:EcUtStatus)`

Privilege: Public
Inherited From: EcPfGenProcess
Constructor which will set attributes from configuration file. It will then read arguments from the command line to reset attributes dynamically.

`EcUtStatus PfCheckStrOfInt (RWCString*)`

Privilege: Protected Operation
Inherited From: EcPfGenProcess
Check input data type for integer

`EcTInt PfGetAppID (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the application ID

`RWCString PfGetAppName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains application name

`RWCString PfGetConfigFileName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the full path name of the configuration file

`EcPfConfigFile* PfGetConfigFileP (EcUtStatus*)`

Privilege: Protected Operation
Inherited From: EcPfGenProcess
Obtains the pointer to the configuration file

`RWCString PfGetDeltaTime (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains delta time

`RWCString PfGetExecName (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the executable name

`EcTInt PfGetMajorVersion (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the major version of the program

`EcTInt PfGetMinorVersion (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the minor version of the program

`RWCString PfGetMode (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the mode

`EcTInt PfGetPID (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the process ID of the program

`RWCString PfGetPath (server:RWCString, EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Construct the CDS path entry using mySite, myMode and the specified server name

```
RWCString PfGetPath (site:RWCString, server:RWCString,
EcUtStauts*)
```

Privilege: Public
Inherited From: EcPfGenProcess
Construct the CDS path entry using myMode, and the specified site name and server name

```
RWCString PfGetPath (site:RWCString,
server:RWCString,Mode:RWCString,EcUtStatus*)
```

Privilege: Public
Inherited From: EcPfGenProcess
Construct the CDS path entry using the specified Mode, site name and server name

`EcTInt PfGetProgramID (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the program ID

`RWCString PfGetSite (EcUtStatus*)`

Privilege: Public
Inherited From: EcPfGenProcess
Obtains the site name

`void PfProcessErrorMsg (EcLgErrorMsg*)`

Privilege: Public
Inherited From: EcPfGenProcess
TBD

`EcTVoid PfSetAttrFromArgv (arc:EcTInt,argv:EcTChar**)`

Privilege: Protect
Inherited From: EcPfGenProcess
Read argument list from command line and set the attributes

`void ~EcPfGenProcess ()`

Privilege: Public
Inherited From: EcPfGenProcess
Destructor

5.2.2.34 Class EcSbNotification

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

Attributes:

`myEcDcDSyncP`

Privilege: Private
Data Type: EcDcDSync*
Default Value: NOT IDENTIFIED
No Inheritance
A pointer to EcDsDSync class

Operations:

`void EcSbNotification ()`

Privilege: Public
No Inheritance
This is the default constructor

`void Send (RWCString:logicalName , RWCString:emailAdd , RWCString:msgLen , RWCString:msg)`

Privilege: Public
No Inheritance
This operation will be used to send notification message and its corresponding information.

`void ~EcSbNotification ()`

Privilege: Public
No Inheritance
Destructor

5.2.2.35 Class EcSeGSSB

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

This is an abstract class which provides the bulk of the functionality for GSS. A user of this class must derive a class from this one which implements the ReadData and WriteData member functions at a minimum. The EcSeGSSTCPB class is an example of such a derivation and the default class which most people should use.

Attributes:

`actual_mech`

Privilege: Private
Data Type: gss_OID
Default Value: NOT IDENTIFIED
No Inheritance
The underlying mechanism used for authentication. May be either DCE or Kerberos.

`ctx_established`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Indicates whether or not a security context has been established. This attribute is checked by all member functions that require a previously established context. The function will fail if this attribute is not set.

Operations:

`EcUtStatus AcceptSecContext (prncplName:const EcTChar* delCred:gss_cred_id_t*)`

Privilege: Public
No Inheritance
This function is called only by the server process and serves to accept a security context initiated by a client. This must be the first GSS member function called unless the corresponding constructor is used (see constructors).

`EcTInt DelSecContext (ctxHndl:gss_ctx_id_t*)`

Privilege: Public
No Inheritance
This functioin deletes the security context associated with the object. Once this function is called none of the other member functions will work except InitSecContext and AcceptSecContext.

`void EcSeGSSB ()`

Privilege: Public
No Inheritance
Default constructor - This allows a GSS object to be created without establishing a security context. Note, however, that a security context must be established before any of the GSS may be used (see InitSecContext and AcceptSecContext).

```
void EcSeGSSB (serverName:const EcTChar* reqFlags:EcTInt
reqTime:EcTInt)
```

Privilege: Public
No Inheritance
Client Constructor - This constructor is used only by the client when the client wishes to establish a security context upon the creation of the GSS object. If this constructor is used, the InitSecContext function should not be called.

```
void EcSeGSSB (prncplName:const EcTChar*
delCred:gss_cred_id_t*)
```

Privilege: Public
No Inheritance
Server Constructor - This constructor is used only by the server when the server wishes to establish a security context upon the creation of the GSS object. If this constructor is used the AcceptSecContext function should not be called.

`OM_uint32 GetMechs ()`

Privilege: Public
No Inheritance
Returns the value of the actual_mech attribute.

```
gss_buffer_t* GetTextName (inName:const EcTChar*
outName:EcTChar&* nameType:gss_OID*)
```

Privilege: Public
No Inheritance
Returns a textual representation of an opaque internal name.

`long GetTimeLeft (ctxHndl:gss_ctx_id_t)`

Privilege: Public
No Inheritance
Returns the remaining amount of time for which the security context is established.

`EcTInt GssErr (maj_stat:OM_uint32 min_stat:OM_uint32 gssStat:EcUtStatus*)`

Privilege: Private
No Inheritance
This function determines if a DCE GSS error occured and if so, convertes it to an EcUtStatus.

`EcUtStatus InitSecContext (serverName:const EcTChar* reqFlags:EcTInt reqTime:EcTInt)`

Privilege: Public
No Inheritance
This function is called by the client process only and serves to initiate the establishment of a security context. This must be the first GSS member function called unless the corresponding constructor is used (see constructors).

`void RcvData (rcvBuf:EcTVoid&* secLevel:EcTInt*)`

Privilege: Public
No Inheritance
This function will receive a buffer using the virtual ReadData member function, unsecure it and return both the buffer and the security level used to secure the message.

`EcTInt ReadData (buf:EcTVoid* len:unsigned)`

Privilege: Protected Operation
No Inheritance
This is a pure virtual member function for reading a buffer from some communication device. This function is used by InitSecContext, AcceptSecContext, SendData and RcvData.

`void SecureMsg (inbuf:const EcTVoid* outbuf:EcTVoid&* secLevel:EcTInt ctxHndl:gss_ctx_id_t*=NULL)`

Privilege: Public
No Inheritance
This function applies security to an input buffer and returns an opaque version of the buffer without sending the buffer. This function is good for incorporating GSS into existing code and for cases where data must be sent via some means other than the WriteData member function.

`void SendData (sndbuf:const EcTVoid* secLevel:EcTInt ctxHndl:gss_ctx_id_t*)`

Privilege: Public
No Inheritance
This function will apply the specified secuity to the given buffer and send the buffer using the virtual WriteData member function.

`void UnSecureMsg (inbuf:const EcTVoid* outbuf:EcTVoid&* secLevel:EcTInt&)`

Privilege: Public
No Inheritance
This function takes an opaque buffer, unsecures it and returns both the readable buffer and the security level used to secure it. This function is good for incorporating GSS into existing code and for instances where data must be received via some means other that the ReadData member function.

`EcTInt WriteData (buf:const EcTVoid* len:unsigned)`

Privilege: Protected Operation
No Inheritance
This is a pure virtual function for writing a buffer to some communication device. This function is used by InitSecContext, AcceptSecContext, SendData, RcvData.

`void ~EcSeGSSB ()`

Privilege: Public
No Inheritance
Destructor deletes the security context and destroys the object.

5.2.2.36 Class EcSeGSSTCPB

Synopsis:

Parent Class: EcSeGSSB
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcSeGSSB

Description:

This is the concrete derivation of the EcSeGSSB class. This class implements the GSS using TCP sockets. A connection must be established prior the instantiating this object.

Attributes:

`fd`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
This is the socket file descriptor associated with the connection that this object is to use.

`actual_mech`

Privilege: Private
Data Type: gss_OID
Default Value: NOT IDENTIFIED
Inherited From: EcSeGSSB
The underlying mechanism used for authentication. May be either DCE or Kerberos.

`ctx_established`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcSeGSSB
Indicates whether or not a security context has been established. This attribute is checked by all member functions that require a previously established context. The function will fail if this attribute is not set.

Operations:

`void EcSeGSSTCPB (sockfd:EcTInt)`

Privilege: Public
No Inheritance
This constructor instantiates the class using a socket file descriptor from a connection that must already be established. I also calls the default constructor for EcSeGSSB.

`void EcSeGSSTCPB (serverName:const EcTChar* reqFlags:EcTInt reqTime:EcTInt sockfd:EcTInt)`

Privilege: Public
No Inheritance
This constructor initializes this class with the socket file descriptor and then calls the corresponding EcSeGSSB constructor.

`void EcSeGSSTCPB (prncplName:EcTChar* delCred:gss_cred_id_t* sockfd:EcTInt)`

Privilege: Public
No Inheritance
This constructor initializes the socket file descriptor and then calls the corresponding EcSeGSSB constructor.

`EcTInt ReadData (buf:EcTVoid* len:unsigned)`

Privilege: Protected Operation
No Inheritance
This is the TCP socket implementation of the pure virtual ReadData function in EcSeGSSB.

`EcTInt WriteData (buf:const EcTVoid* len:unsigned)`

Privilege: Protected Operation
No Inheritance
This the TCP socket implementation of the pure virtual WriteData function from EcSeGSSB.

`void ~EcSeGSSTCPB ()`

Privilege: Public
No Inheritance
Calls the EcSeGSSB destructor then destroys the object.

`EcUtStatus AcceptSecContext (prncplName:const EcTChar* delCred:gss_cred_id_t*)`

Privilege: Public
Inherited From: EcSeGSSB
This function is called only by the server process and serves to accept a security context initiated by a client. This must be the first GSS member function called unless the corresponding constructor is used (see constructors).

`EcTInt DelSecContext (ctxHndl:gss_ctx_id_t*)`

Privilege: Public
Inherited From: EcSeGSSB
This functioin deletes the security context associated with the object. Once this function is called none of the other member functions will work except InitSecContext and AcceptSecContext.

`void EcSeGSSB ()`

Privilege: Public
Inherited From: EcSeGSSB
Default constructor - This allows a GSS object to be created without establishing a security context. Note, however, that a security context must be established before any of the GSS may be used (see InitSecContext and AcceptSecContext).

`void EcSeGSSB (serverName:const EcTChar* reqFlags:EcTInt reqTime:EcTInt)`

Privilege: Public
Inherited From: EcSeGSSB
Client Constructor - This constructor is used only by the client when the client wishes to establish a security context upon the creation of the GSS object. If this constructor is used, the InitSecContext function should not be called.

`void EcSeGSSB (prncplName:const EcTChar* delCred:gss_cred_id_t*)`

Privilege: Public
Inherited From: EcSeGSSB
Server Constructor - This constructor is used only by the server when the server wishes to establish a security context upon the creation of the GSS object. If this constructor is used the AcceptSecContext function should not be called.

`OM_uint32 GetMechs ()`

Privilege: Public
Inherited From: EcSeGSSB
Returns the value of the actual_mech attribute.

`gss_buffer_t* GetTextName (inName:const EcTChar* outName:EcTChar&* nameType:gss_OID*)`

Privilege: Public
Inherited From: EcSeGSSB
Returns a textual representation of an opaque internal name.

`long GetTimeLeft (ctxHndl:gss_ctx_id_t)`

Privilege: Public
Inherited From: EcSeGSSB
Returns the remaining amount of time for which the security context is established.

```
EcTInt GssErr (maj_stat:OM_uint32 min_stat:OM_uint32
gssStat:EcUtStatus*)
```

Privilege: Private
Inherited From: EcSeGSSB
This function determines if a DCE GSS error occured and if so, convertes it to an EcUtStatus.

```
EcUtStatus InitSecContext (serverName:const EcTChar*
reqFlags:EcTInt reqTime:EcTInt)
```

Privilege: Public
Inherited From: EcSeGSSB
This function is called by the client process only and serves to initiate the establishment of a security context. This must be the first GSS member function called unless the corresponding constructor is used (see constructors).

```
void RcvData (rcvBuf:EcTVoid&* secLevel:EcTInt*)
```

Privilege: Public
Inherited From: EcSeGSSB
This function will receive a buffer using the virtual ReadData member function, unsecure it and return both the buffer and the security level used to secure the message.

```
EcTInt ReadData (buf:EcTVoid* len:unsigned)
```

Privilege: Protected Operation
Inherited From: EcSeGSSB
This is a pure virtual member function for reading a buffer from some communication device. This function is used by InitSecContext, AcceptSecContext, SendData and RcvData.

```
void SecureMsg (inbuf:const EcTVoid* outbuf:EcTVoid&*
secLevel:EcTInt ctxHndl:gss_ctx_id_t*=NULL)
```

Privilege: Public
Inherited From: EcSeGSSB
This function applies security to an input buffer and returns an opaque version of the buffer without sending the buffer. This function is good for incorporating GSS into existing code and for cases where data must be sent via some means other than the WriteData member function.

```
void SendData (sndbuf:const EcTVoid* secLevel:EcTInt
ctxHndl:gss_ctx_id_t*)
```

Privilege: Public
Inherited From: EcSeGSSB
This function will apply the specified secuity to the given buffer and send the buffer using the virtual WriteData member function.

```
void UnSecureMsg (inbuf:const EcTVoid* outbuf:EcTVoid&*
secLevel:EcTInt&)
```

Privilege: Public
Inherited From: EcSeGSSB
This function takes an opaque buffer, unsecures it and returns both the readable buffer and the security level used to secure it. This function is good for incorporating GSS into existing code and for instances where data must be received via some means other that the ReadData member function.

```
EcTInt WriteData (buf:const EcTVoid* len:unsigned)
```

Privilege: Protected Operation
Inherited From: EcSeGSSB
This is a pure virtual function for writing a buffer to some communication device. This function is used by InitSecContext, AcceptSecContext, SendData, RcvData.

```
void ~EcSeGSSB ()
```

Privilege: Public
Inherited From: EcSeGSSB
Destructor deletes the security context and destroys the object.

5.2.2.37 Class EcSeServerKeyMgmt

Synopsis:

Parent Class: DCEPassword
Is Not A Distributed Object
Is Associated With:
Class: rgy_edit(Private) Creates&ProvidesserverKeyFile
Class: appServerObj(Private) ProvidesappServerObjPasswordintheserverKeyFile

Description:

This class is the concrete implementation of the abstract DCEPassword class using a file as the means of retrieving the secret key. This class provides a consistent way for accessing the password data for a particular security principal (non interactive principal - server).

Attributes:

`_keyFile`

Privilege: Protected Attribute
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents the server keytab file where the server's password is encoded.

`_pName`

Privilege: Protected Attribute
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
This is the principal name whose password is stored in the keytab file.

`_passwordValid`

Privilege: Protected Attribute
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents a flag which indicates whether a password is valid or invalid.

`_passwordValidMutex`

Privilege: Protected Attribute
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents the pthread mutex for the _passwordValid data member.

Operations:

```
void EcSeServerKeyMgmt (a_pName:const RWCString&
a_localKeyFile:const RWCString&)
```

Privilege: Public
No Inheritance
This is the class constructor. It makes a copy of the supplied principalName and keyFileName into respective class variables.

`sec_passwd_rec_t* GetPassword ()`

Privilege: Public
No Inheritance
This operation is the implementation of the base class pure virtual making a dce call to retrieve the secret key of a principal from the _keyFile.

`EcTInt GetPasswordValid (status:EcUtStatus&)`

Privilege: Private
No Inheritance
This method will get the value of the password validity(0 or 1).

`EcUtStatus SetPasswordValid (pswdValid:EcTInt)`

Privilege: Private
No Inheritance
This member function will set the password validity to 0 or 1.

`EcTVoid invalidatePassword ()`

Privilege: Public
No Inheritance
Renders a password not valid.

`void ~EcSeServerKeyMgmt ()`

Privilege: Public
No Inheritance
This is the class destructor.

5.2.2.38 Class EcTiTimeService

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcFosTimeProviderB(Private) Interactswith

Description:

This class is used to obtain the current time in various formats.

Attributes:

`_delta_indicator`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Indicates what to do with the _delta_value; 0 = Obtain current time; 1 = Add delta to the current time.

`_delta_value`

Privilege: Private
Data Type: utc_t
Default Value: NOT IDENTIFIED
No Inheritance
Delta that will be added to the current time.

Operations:

`EcUtStatus AddTime (a_utc1 const utc_t a_utc2 :const utc_t ao_result :utc_t&)`

Privilege: Public
No Inheritance
This method adds two binary timestamps, producing a third binary timestamp whose inaccuracy is the sum of the two input inaccuracies.

`EcUtStatus ApplyDelta (a_utc :const utc_t& ao_utc :utc_t&)`

Privilege: Public
No Inheritance
Apply the delta to the current time

`EcUtStatus BinToAscGmt (a_utc :const utc_t& ao_TimeString[] :EcTChar)`

Privilege: Private
No Inheritance

`EcUtStatus CalculateDelta (a_testtime :const utc_t&)`

Privilege: Public
No Inheritance
Given an absolute time calculate the delta.

`EcUtStatus CmpIntTime (a_utc1 :const utc_t& a_utc2 :const utc_t& ao_relation :enum utc_cmptype&)`

Privilege: Public
No Inheritance
Compares two binary timestamps or two relative binary timestamps and returns the relationship not ignoring inaccuracies.

`EcUtStatus CmpMidTime (a_utc1 :const utc_t& a_utc2 :const utc_t& ao_relation :enum utc_cmptype&)`

Privilege: Public
No Inheritance

```
EcUtStatus CvtDeltaToBinary (a_deltastring :const EtcChar*)
```

Privilege: Public
No Inheritance
Convert a delta value from the namespace to a binary timestamp.

```
EcUtStatus CvtStrToBin (a_TimeString :const EcTChar* ao_utc
utc&)
```

Privilege: Private
No Inheritance
Converts character string to binary timestamp.

```
void EcTiTimeService ()
```

Privilege: Public
No Inheritance
Default constructor. When a NULL string is passed as a parameter in the constructor,the current time will be obtained.

```
void EcTiTimeService (status :EcUtStatus* a_NameSpace :const
EcTChar* a_DeltaType :EcTInt)
```

Privilege: Public
No Inheritance
Constructor. Creates an instance of the class ,given a name from the namespace and a flag that indicates how delta is passed (as a relative delta or incorporated in the absolute testtime).

```
EcUtStatus GetAscGmtTime (ao_TimeString[] :EcTChar)
```

Privilege: Public
No Inheritance
Obtain current GMT time in ASCII string format.

```
EcUtStatus GetAscGmtTime (ao_TimeString :RWCString)
```

Privilege: Public
No Inheritance
Obtain current GMT Time in RWCString format

```
 GetLocalTime (a_utc :const utc_t& ao_timetm :struct tm&
ao_tns :EcTLongInt& ao_inacctm :struct tm& ao_ins
:EcTLongInt&)
```

Privilege: Protection Not Identified
No Inheritance
Converts a binary timestamp to a tm structure that expresses local time.

`EcUtStatus GetLocalZone (a_utc :const utc_t& a_tzlen :EcTInt& ao_tzname[] :EcTChar ao_tdf :EcTLongInt& ao)`

Privilege: Public
No Inheritance
This method gets the local time zone label and offset from GMT, given utc.

`EcUtStatus GetSecNanoTime (ao_timesp :timespec_t& ao_inaccsp :timespec_t& ao_tdf :EcTLongInt&)`

Privilege: Public
No Inheritance
Obtain current time (timespec_t - seconds, nanoseconds)

`EcUtStatus GetTime (ao_utc utc_t&)`

Privilege: Private
No Inheritance
Obtains current binary timestamp.

`EcUtStatus GetTimeValues (ao_tm : struct tm&)`

Privilege: Public
No Inheritance
Obtain current time (tm structure - seconds, minutes, hours, day of month, month of year, year, day of week, day of week, day of year, flag for daylight savings time).

`EcUtStatus GmTime (a_utc :const utc_t& ao_tm :struct tm&)`

Privilege: Public
No Inheritance
Converts a Binary stamp to a TM structure.

`EcUtStatus MkBinRelTime (a_timesp :const timespec_t a_iaccsp :const timespec_t& ao_utc :utc_t&)`

Privilege: Public
No Inheritance
Make Binary Relative Time from the timespec_t structure

`EcUtStatus MkBinTime (a_timesp :const timespec_t& a_inaccsp :const timespec_t& a_tdf :EcTLongInt& ao_u)`

Privilege: Public
No Inheritance
This method converts a timespec structure time to a binary timestamp.

`EcUtStatus MkGMTTime (a_timetm :const struct tm& a_tns :EcTLongInt ao_utc :etc_t)`

Privilege: Public
No Inheritance
This method converts a trn structure that expresses GMT or UTC to a binary stamp.

`EcUtStatus SubTime (a_utc1 :const utc_t& a_utc2 :const utc_t& ao_result :utc_t&)`

Privilege: Public
No Inheritance
The method subtracts one time stamp from another.

`void ~EcTiTimeService ()`

Privilege: Public
No Inheritance
Destructor for the EcTiTimeService Class.

5.2.2.39 Class EcUrClassID

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
GrLiAnyURClass (Aggregation)

Description:

This class encapsulates an indentifier for ECS C++ classes. It supports several constructors, a mechanism for comparing two instance of this class, and the ability to read/write itself from streams. A const global object of this class, "theInvalidClassID" is defined. It can be used to set/check if something is out of range.

Attributes:

`myRep`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
This attribute is the internal representation of the Class ID.

Operations:

`void EcUrClassID (void)`

Privilege: Public
No Inheritance
This method is the default constructor for this class. This object's value, while initialized, is still undefined.

`void EcUrClassID (const int)`

Privilege: Public
No Inheritance
This method will construct an object whose value is defined by the argument.

`void EcUrClassID (const char*)`

Privilege: Public
No Inheritance
This method will construct an object whose value is defined by the argument.

`void EcUrClassID (const EcUrClassID&)`

Privilege: Public
No Inheritance
This method is the copy constructor the object.

`EcTBoolean IsValid (void)`

Privilege: Public
No Inheritance

`EcTUInt hash (void)`

Privilege: Public
No Inheritance

`int operator!= (const EcUrClassID&)`

Privilege: Public
No Inheritance
This method is the inequality operator. It will return a non-zero value if the objects being compared are logically different.

`ostream& operator<< (ostream&)`

Privilege: Public
No Inheritance

`int operator== (const EcUrClassID&)`

Privilege: Public
No Inheritance

```
istream& operator>> (istream&)
```

Privilege: Public
No Inheritance
This is the stream extraction operator for the method. Unfortunately, while the name 'extraction' is similar to the UR operation 'Extract' its behavior is unrelated. This method reads the contents of the class ID from the input stream argument. It can only read a class ID written with the insertion operator<<. An exception will be raised if the stream or stream data is invalid.

```
void ~EcUrClassID (void)
```

Privilege: Public
No Inheritance
This method is the destructor for the object. State cleanup before the object is destroyed will occur here.

5.2.2.40 Class EcUrUR

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcUrURMaker(Public)
Class: EcUrURProvider(Public)

Description:

This is the abstract base class for all Universal Reference (UR)s. A UR is a special ECS identifier for an object. What makes it special is that an object can be identified, but the object does not have to exist in memory at the time. The contents of a UR are specified by subclasses. Generally speaking, the contents are the key elements of the object that this UR refers to. It can be thought of as DNA. We can reconstitute or clone an organism (i.e. object or URProvider) given its DNA (i.e. UR). The key public methods are "Externalize" and "Internalize"

Attributes:

Operations:

```
void EcUrUR (void)
```

Privilege: Protected Operation
No Inheritance
This protected constructor initializes the object.

`void Externalize (ostream&)`

Privilege: Public
No Inheritance
This method is the public interface for exporting the contents of the UR to an output stream. This data can be imported later with Internalize. This method allows URs to persist outside the runtime of the application. The stream will contain information (processed by Internalize) that prevents tampering.

`void ExternalizeClassData (ostream&)`

Privilege: Protected Operation
No Inheritance
This method exports the UR's state to the output stream argument. This method shall be overridden by all concrete UR classes. The data written should have been read back from the stream with InternalizeClassData. This method shall call all appropriate associated objects' ExternalizeClassData method. An associated object is either a direct base class or a contained object. Note the public interface for this functionality is Externalize.

`const EcUrClassID& GetURID (void)`

Privilege: Public
No Inheritance
This method returns the class ID of the UR object. Often abstract base class URs are passed around. This method can be used to find the actual conrete object being passed. The method GetURProviderID can be used to findout what concrete object this UR refers to.

`const EcUrClassID& GetURProviderID (void)`

Privilege: Public
No Inheritance
The method returns the Class ID of the object that is referred to by this UR. This is the object that Extract'ed this UR and that can Reconstitute itself from this UR.

`void Internalize (istream&)`

Privilege: Public
No Inheritance
This is the public method for importing data from a stream into a UR. If the stream does not contain the correct kind of data for this UR or if the data is invalid, an exception will be raised.

`void InternalizeClassData (istream&)`

Privilege: Protected Operation
No Inheritance
This method imports the UR's state from the input stream argument. This method shall be overridden by all concrete UR classes. The data read should have been written to the stream with ExternalizeClassData. This method shall call all appropriate associated objects InternalizeClassData method. An associated object is either a direct base class or a contained object. Note the public interface for this functionality is Internalize. Exceptions should be raised in this operation if errors occur.

`void ReadTypingData (istream&, EcUrClassID&)`

Privilege: Private
No Inheritance
This private method allows a friend object class to determine the UR class ID contained in a stream. It is used by URMaker. The class ID is placed in the Class ID reference argument.

`ostream& operator<< (ostream&)`

Privilege: Public
No Inheritance
This method the same as the Externalize method except it support the standard stream insertion signature.

`istream& operator>> (istream&)`

Privilege: Public
No Inheritance
This method the same as the Internalize method except it support the standard stream extraction signature.

`void ~EcUrUR (void)`

Privilege: Protected Operation
No Inheritance
This protected method is the destructor for the object. It will clean up state prior to the object being destroyed.

5.2.2.41 Class EcUrURMaker

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcUrUR(Public)

Description:

This class supports two correlated responsibilities. First, it is an object factory for Universal Reference (UR)s. It allows subclasses of URs to register themselves. Then based on a given encapsulated ClassID, it can dynamically construct URs of any registered type. Secondly, it can decode a stream containing externalized (i.e. ASCII represented) URs. This class can read a stream containing a UR and identify the UR specified in the stream or the UR Provider referred to by the UR in the stream.

Attributes:

`myStream`

Privilege: Private
Data Type: istream&
Default Value: NOT IDENTIFIED
No Inheritance
This attribute is a pointer to the current input stream associated with this object. This stream contains exported URs that this object helps to import.

Operations:

`void DeleteUR (const EcUrUR*)`

Privilege: Public
No Inheritance
This static function is provided as an aid to callers who used MakeUR to create a "const EcUR*. While they are responsible for deleting it, they can't because it is const. This routine can delete it.

`void EcUrURMaker (void)`

Privilege: Public
No Inheritance
This method is the default constructor for the object. The stream still must be set after this constructor.

`void EcUrURMaker (const EcUrURMaker&)`

Privilege: Public
No Inheritance

`const EcClassID& GetURID (istream&)`

Privilege: Public
No Inheritance
This method returns a reference to the Class ID of the exported UR currently at the beginning of the input stream. The returned class ID will be invalid if the stream does not contain a valid UR.

`const EcClassID& GetURProviderID (istream&)`

Privilege: Public
No Inheritance
This method returns a reference to the Class ID of the UR Provider object referred to by the exported UR currently at the beginning of the input stream. An invalid class ID will be returned if the stream does not contain a valid UR.

`const EcUR* MakeUR (istream&)`

Privilege: Public
No Inheritance
Makes a UR and Internalize it. The user is responsible for deleting the return value through ::DeletedUR. This function can return NULL if the UR defined in the stream is not registered.

` Register (const EcUtClassID&, EcUrURProvider*(*func)(), EcTBoolean replaceOK=FALSE)`

Privilege: Protection Not Identified
No Inheritance

`void ~EcUrURMaker (void)`

Privilege: Public
No Inheritance
This method is the destructor for the object. It will clean up the state to allow proper deallocation. The state of the internal stream will not be affected by this operation.

5.2.2.42 Class EcUrURProvider

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcUrUR(Public)
Class: EcUrURProviderMaker(Public)

Description:

This class is the abstract base class for all things refered to by Universal Reference (UR)s. Its primary responsibility is to provide URs to clients, thus the name "UR Provider". The primary operations of interest are "ProvideUR" and "Reconstitute".

Attributes:

Operations:

`EcUrUR* CreateUR (void)`

Privilege: Protected Operation
No Inheritance
This method will create and return an new UR that can refer to this object. This method shall be overridden by all concrete derived classes of this class.

`void DeleteUR (const EcUrUR*)`

Privilege: Public
No Inheritance
This function is static. Since we return a const UR, from ProvideUR, the client can't delet it. This method is responsible for deleting it.

`void EcUrURProvider (void)`

Privilege: Protected Operation
No Inheritance
This method is the constructor for the class.

`static const EcUrClassID& GetMyClassID (void)`

Privilege: Public
No Inheritance

`void ProvideClassUR (EcUrUR&)`

Privilege: Protected Operation
No Inheritance
Provide primary key data for the class state and place it in the UR. Then, call this method for each of your associated objects.

`const EcUrUR* ProvideUR (void)`

Privilege: Public
No Inheritance
This method will provide a Universal Reference to the caller that represents the current logical entity. The return value is allocated on the heap and should be deallocated with the "DeleteUR" method. The return value is a "Memento" (standard design pattern) that can be used to logically bring this object back. The "Reconstitute" method can be called to bring an object back to this state. Note that state is application specific. Derived classes should decide on policies. Possibilities include bring back that exact object the UR came from or reconstituting to the latest version.   This method is a "template method" (standard design pattern) that calls "ProvideClassUR".

```
void Reconstitute (const EcUrUR&)
```

Privilege: Public
No Inheritance
Public method to make ourselves the object that is logically referred to by the UR.

```
void ReconstituteClassData (const EcUrUR&)
```

Privilege: Protected Operation
No Inheritance
Reconstitute class data self based on the UR. Then, call this for associated objects.

```
void ~EcUrURProvider (void)
```

Privilege: Protected Operation
No Inheritance
This method is the destructor for the class.

5.2.2.43 Class EcUrURProviderMaker

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: EcUrURProvider(Public)

Description:

This class is an object factory responsible for the registration and dynamic creation of object subclasses from "URProvider". Objects are indexed by the encapsulated type "ClassID".

Attributes:

Operations:

```
void EcUrURProviderMaker (void)
```

Privilege: Public
No Inheritance
This method is the constructor for this object.

```
EcUrURProvider* MakeURProvider (const EcUR&)
```

Privilege: Public
No Inheritance
Make a URProvider that matches the UR argument and then Reconstitute it.

`void Register (const EcClassID&, EcURProvider* (*func)(), EcTBoolean replaceOK=FALSE)`

Privilege: Public
No Inheritance
Register a creation function for a UR Provider derived object.

`void ~EcUrURProviderMaker (void)`

Privilege: Public
No Inheritance
This method is the destructor for this method.

5.2.2.44 Class EcUtLoggerRelA

Synopsis:

Parent Class: strstream
Distributed Object
Is Associated With:
This class is derived from the class strstream

Description:

This service allows applications to log event and history information to a file which can later be used for study.

Attributes:

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
No Inheritance
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
No Inheritance
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
No Inheritance
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
No Inheritance
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
No Inheritance
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
No Inheritance
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
No Inheritance
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
No Inheritance
will be used to hold the os name and version

`ourSynchMutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
No Inheritance
mutex to protect internal integrity.

Operations:

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
No Inheritance
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
No Inheritance
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
No Inheritance
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
No Inheritance
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
No Inheritance
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
No Inheritance
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
No Inheritance
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
No Inheritance
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
No Inheritance
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
No Inheritance
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
No Inheritance
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
No Inheritance
- internal function - locks file

```
void ~EcUtLoggerRelA ()
```

Privilege: Public
No Inheritance
Destructor

5.2.2.45 Class EcUtLoggerRelAAudit

Synopsis:

Parent Class: EcUtLoggerRelAMgmt
Distributed Object
Is Associated With:
This class is derived from the class EcUtLoggerRelAMgmt

Description:

Used to create Audit event logging objects.

Attributes:

```
myDisposition
```

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the disposition

```
myEventType
```

Privilege: Protected Attribute
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the event type

```
myId
```

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold id

```
myLoggingStatus
```

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Logging status

`myParentId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Will hold parent id

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcUtLoggerRelA
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the os name and version

```
ourSynchMutex
```

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
mutex to protect internal integrity.

Operations:

```
void EcUtLoggerRelAAudit (appname:EcTChar *
appversion:EctChar * = NULL)
```

Privilege: Public
No Inheritance
constructor - accepts application name and an optional application version as arguments.

```
EcTVoid SetMyEventType (EcTInt)
```

Privilege: Public
No Inheritance
Sets the event type.

```
void ~EcUtLoggerRelAAudit ()
```

Privilege: Public
No Inheritance
destructor

```
void EcUtLoggerRelAMgmt (appname: EctChar *
appversion:EcTChar * = NULL)
```

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
constructor - accepts as arguments, the application name and an opional version.

```
void SetMyEventType (EcTInt)
```

Privilege: Protected Operation
Inherited From: EcUtLoggerRelAMgmt
Used to set the event type.

```
void SetMyId (:EcTChar *)
```

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the id

`void SetMyParentId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the parent id

`void ~EcUtLoggerRelAMgmt ()`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
destructor

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
Inherited From: EcUtLoggerRelA
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`void ~EcUtLoggerRelA ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
Destructor

5.2.2.46 Class EcUtLoggerRelADebug

Synopsis:

Parent Class: EcUtLoggerRelA
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcUtLoggerRelA

Description:

Used to log application debugging information.

Attributes:

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcUtLoggerRelA
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the os name and version

`ourSynchMutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
mutex to protect internal integrity.

Operations:

`void EcUtLoggerRelADebug (appname:EcTChar * appversion:EcTChar * = NULL)`

Privilege: Public
No Inheritance
Constructor. Accepts as arguments the application name and an optional version string.

`EcTBoolean GetLoggingStatus ()`

Privilege: Public
No Inheritance
Will return if logging is currently on or off.

`EcTVoid SetLoggingStatus (:EctBoolean)`

Privilege: Public
No Inheritance
This will turn on and off logging of messages.

`void ~EcUtLoggerRelADebug ()`

Privilege: Public
No Inheritance
Destructor

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
Inherited From: EcUtLoggerRelA
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`void ~EcUtLoggerRelA ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
Destructor

5.2.2.47 Class EcUtLoggerRelAFault

Synopsis:

Parent Class: EcUtLoggerRelAMgmt
Distributed Object
Is Associated With:
This class is derived from the class EcUtLoggerRelAMgmt

Description:

Used to log Fault events.

Attributes:

`myDisposition`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the disposition

`myEventType`

Privilege: Protected Attribute
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the event type

`myId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold id

`myLoggingStatus`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Logging status

`myParentId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Will hold parent id

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcUtLoggerRelA
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the os name and version

`ourSynchMutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
mutex to protect internal integrity.

Operations:

`void EcUtLoggerRelAFault (appname:EcTChar * appversion:EcTChar * = NULL)`

Privilege: Public
No Inheritance
constructor - accepts application name and an optional application version as arguments.

`EcTVoid SetMyEventType (EcTInt)`

Privilege: Public
No Inheritance
Used to set the event type.

`void ~EcUtLoggerRelAFault ()`

Privilege: Public
No Inheritance
destructor

`void EcUtLoggerRelAMgmt (appname: EctChar * appversion:EcTChar * = NULL)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
constructor - accepts as arguments, the application name and an opional version.

`void SetMyEventType (EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelAMgmt
Used to set the event type.

`void SetMyId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the id

`void SetMyParentId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the parent id

`void ~EcUtLoggerRelAMgmt ()`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
destructor

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
Inherited From: EcUtLoggerRelA
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`void ~EcUtLoggerRelA ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
Destructor

5.2.2.48 Class EcUtLoggerRelAMgmt

Synopsis:

Parent Class: EcUtLoggerRelA
Distributed Object
Is Associated With:
This class is derived from the class EcUtLoggerRelA

Description:

Virtual object for management logging.

Attributes:

`myDisposition`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
will hold the disposition

`myEventType`

Privilege: Protected Attribute
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
will hold the event type

`myId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
No Inheritance
will hold id

`myLoggingStatus`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
Logging status

`myParentId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
No Inheritance
Will hold parent id

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcUtLoggerRelA
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the os name and version

`ourSynchMutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
mutex to protect internal integrity.

Operations:

`void EcUtLoggerRelAMgmt (appname: EctChar * appversion:EcTChar * = NULL)`

Privilege: Public
No Inheritance
constructor - accepts as arguments, the application name and an opional version.

`void SetMyEventType (EcTInt)`

Privilege: Protected Operation
No Inheritance
Used to set the event type.

`void SetMyId (:EcTChar *)`

Privilege: Public
No Inheritance
used to set the id

`void SetMyParentId (:EcTChar *)`

Privilege: Public
No Inheritance
used to set the parent id

`void ~EcUtLoggerRelAMgmt ()`

Privilege: Public
No Inheritance
destructor

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
Inherited From: EcUtLoggerRelA
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`void ~EcUtLoggerRelA ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
Destructor

5.2.2.49 Class EcUtLoggerRelAPerf

Synopsis:

Parent Class: EcUtLoggerRelAMgmt
Distributed Object
Is Associated With:
This class is derived from the class EcUtLoggerRelAMgmt

Description:

Object for logging Performance events.

Attributes:

`myDisposition`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the disposition

`myEventType`

Privilege: Protected Attribute
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the event type

`myId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold id

`myLoggingStatus`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Logging status

`myParentId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Will hold parent id

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcUtLoggerRelA
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the os name and version

`ourSynchMutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
mutex to protect internal integrity.

Operations:

`void EcUtLoggerRelAPerf (appname:EcTChar * appversion:EctChar * = NULL)`

Privilege: Public
No Inheritance
Constructor.

`void SetMyEventType (:EcTInt)`

Privilege: Public
No Inheritance
Set the event type.

`void ~EcUtLoggerRelAPerf ()`

Privilege: Public
No Inheritance
destructor

`void EcUtLoggerRelAMgmt (appname: EctChar * appversion:EcTChar * = NULL)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
constructor - accepts as arguments, the application name and an opional version.

`void SetMyEventType (EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelAMgmt
Used to set the event type.

`void SetMyId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the id

`void SetMyParentId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the parent id

`void ~EcUtLoggerRelAMgmt ()`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
destructor

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
Inherited From: EcUtLoggerRelA
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

```
void ~EcUtLoggerRelA ()
```

Privilege: Public
Inherited From: EcUtLoggerRelA
Destructor

5.2.2.50 Class EcUtLoggerRelASec

Synopsis:

Parent Class: EcUtLoggerRelAMgmt
Distributed Object
Is Associated With:
This class is derived from the class EcUtLoggerRelAMgmt

Description:

Used to log Security events.

Attributes:

`myDisposition`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the disposition

`myEventType`

Privilege: Protected Attribute
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold the event type

`myId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
will hold id

`myLoggingStatus`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Logging status

`myParentId`

Privilege: Private
Data Type: EcTChar[130]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelAMgmt
Will hold parent id

`myAppName`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the application name

`myAppVersion`

Privilege: Private
Data Type: EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the version number

`myErrorBitmask`

Privilege: Private
Data Type: EcTInt32
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ErrorLevel bitmask. It will determine which error levels will get logged.

`myErrorLevel`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: EcUtLoggerRelA
will be used to hold the currently set error level this has a default level of 0

`myLockCount`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used in conjunction with the thread writing mutex to keep a count of locks to the recursive mutex.

`myLogFile`

Privilege: Private
Data Type: EcTChar[EcDMaxFileName]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the name of the log file.

`myMaxFileSize`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
Will hold the maximum file size of the file.

`ourCellName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the cell name.

`ourIPAddress`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the ip address of the local machine

`ourOSName`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will hold the OS name

`ourOSVersion`

Privilege: Private
Data Type: static EcTChar[EcDStr]
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
will be used to hold the os name and version

`ourSynchMutex`

Privilege: Private
Data Type: DCEPthreadMutex
Default Value: NOT IDENTIFIED
Inherited From: EcUtLoggerRelA
mutex to protect internal integrity.

Operations:

`void EcUtLoggerRelASec (appname:EcTChar * appversion:EctChar * = NULL)`

Privilege: Public
No Inheritance
constructor - accepts application name and an optional application version as arguments.

`EcTVoid SetMyEventType (EcTInt)`

Privilege: Public
No Inheritance
Used to set the event type.

`void ~EcUtLoggerRelASec ()`

Privilege: Public
No Inheritance
destructor

`void EcUtLoggerRelAMgmt (appname: EctChar * appversion:EcTChar * = NULL)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
constructor - accepts as arguments, the application name and an opional version.

`void SetMyEventType (EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelAMgmt
Used to set the event type.

`void SetMyId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the id

`void SetMyParentId (:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
used to set the parent id

`void ~EcUtLoggerRelAMgmt ()`

Privilege: Public
Inherited From: EcUtLoggerRelAMgmt
destructor

`void EcUtLoggerRelA (appname: EcTChar * appversion:EcTChar *)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Constructor. Will use the appname to determine the filename to log to. The appversion is optional or may be NULL.

`EcTVoid FreeLock ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - frees lock

`const EcTChar * GetLogFileName ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the log file name to the application. No guarantee is made about the format of the file.

`EcTInt32 GetLoggingBitmask ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will return the bitmask of which errors are currently being logged.

`void HandleMessage (:EcTInt)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Used to handle control messages

`EcTVoid LockDown ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks down structures

`EcTInt LockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`EcTInt OpenFile ()`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
- internal function - opens file

`void Operator<< (:object)`

Privilege: Public
Inherited From: EcUtLoggerRelA
will be used to send messages to the object. Chaining of <<s is encouraged.

`void SetAppName (:EcTChar *)`

Privilege: Protected Operation
Inherited From: EcUtLoggerRelA
Will set the current application - and will as a result have its log file determined.

`void SetErrorLevel (:EcTInt)`

Privilege: Public
Inherited From: EcUtLoggerRelA
Will set the error level for the current message.

`EcTInt UnLockFile ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
- internal function - locks file

`void ~EcUtLoggerRelA ()`

Privilege: Public
Inherited From: EcUtLoggerRelA
Destructor

5.2.2.51 Class Pthread

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: PthreadMutex(Public) can_be_controlled_by

Description:

A Pthread is the representation of a thread of execution. A DCEPthread object corresponds to a single thread. The DCEPthread class provides limited information about a thread and limited control of that thread. A DCEPthread object represents the thread before, during, and after its execution. The thread may also continue to execute after the DCEPthread object has been deleted.

Attributes:

`p`

Privilege: Private
Data Type: PthreadPrio
Default Value: Pthread_pri_mid
No Inheritance
Priority to run the thread(Pthread_pri_min, Pthread_pri_low, Pthread_pri_mid, Pthread_pri_high, Pthread_pri_max).

`param`

Privilege: Private
Data Type: ThreadParam
Default Value: NOT IDENTIFIED
No Inheritance
Parameter passed to thread when thread is started

`proc`

Privilege: Private
Data Type: ThreadProc
Default Value: NOT IDENTIFIED
No Inheritance
Procedure to be executed when thread is started

`s`

Privilege: Private
Data Type: PthreadSched
Default Value: Pthread_fg
No Inheritance
Scheduling type(Pthread_fifo- first in first out, Pthread_rr - round robin, Pthread_fg - foreground non portable, Pthread_bg - background non portable)

`size`

Privilege: Private
Data Type: EcTLong
Default Value: NOT IDENTIFIED
No Inheritance
Size of the stack for the thread

`t`

Privilege: Private
Data Type: PthreadTermination
Default Value: NOT IDENTIFIED
No Inheritance
Thread termination type. (Pthread_no_detach_on_delete, Pthread_detach_on_delete, or Pthread_join_on_delete)

Operations:

`EcTVoid Cancel ()`

Privilege: Public
No Inheritance
Sends a cancel signal to the thread. This operation is only valid once a Pthread has been Start'ed. The thread may block cancellation, so it need not stop immediately.

`ThreadResult Join ()`

Privilege: Public
No Inheritance
Waits for completion of thread.

`void Priority ()`

Privilege: Public
No Inheritance
Retrieves priority

`PthreadPrio Priority (p:PthreadPrio)`

Privilege: Public
No Inheritance
Sets the priority of the thread

`void Pthread ()`

Privilege: Public
No Inheritance
Default constructor

`void Pthread (proc:ThreadProc param:ThreadParam)`

Privilege: Public
No Inheritance
Constructors for the Pthread class which will contruct a Pthread based on default attributes. The second form will also start the thread.

`void Scheduling ()`

Privilege: Public
No Inheritance
Retrieves scheduling algorithm

`PthreadSched Scheduling (s:PthreadSched p:PthreadPrio)`

Privilege: Public
No Inheritance
Sets scheduling and relative priority

`void Stacksize ()`

Privilege: Public
No Inheritance
Retrieves stack size

`EcTLong Stacksize (size:long)`

Privilege: Public
No Inheritance
Sets thread stack size

`EcTVoid Start (proc:ThreadProc param:ThreadParam)`

Privilege: Public
No Inheritance
Launches that thread by executing the specified procedure, passing it the specified parameter. This operation may only be performed on a Pthread that has no been previously Start'ed.

```
void Termination ()
```

Privilege: Public
No Inheritance
Retrieves termination options

```
void Termination (t:PthreadTermination)
```

Privilege: Public
No Inheritance
Sets termination options

```
void ~Pthread ()
```

Privilege: Public
No Inheritance
The destructor cleans up the Pthread data.

5.2.2.52 Class PthreadCond

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

The PthreadCond class encapsulates the pthread_cond_t condition type. A mutex is associated with the PthreadCond when it is constructed.

Attributes:

Operations:

```
void Broadcast ()
```

Privilege: Public
No Inheritance
This call signals a thread waiting on the condition variable that the condition may now be true. Broadcast wakes up all the threads that are waiting on the condition variable.

```
void PthreadCond (x:PthreadMutex&)
```

Privilege: Public
No Inheritance
The only constructor for PthreadCond takes a mutex as a parameter. This is the mutex associated with the condition. This mutex must be locked before certain operations may take place.

`void Signal ()`

Privilege: Public
No Inheritance
This call signals a thread waiting on the condition variable that the condition may now be true. Signal wakes up one thread that is waiting on the condition variable.

`void Wait ()`

Privilege: Public
No Inheritance
This operation initiates a wait on the condition variable. The caller should have set the mutex, then tested for the desired condition before making the call. A return implies that the mutex is now set for the caller. The condition must be re-checked before continuing. If the condition is still not true, the caller may wish to wait again.

`void Wait (t:PthreadTime)`

Privilege: Public
No Inheritance
This operation initiates a wait on the condition variable. The caller should have set the mutex, then tested for the desired condition before making the call. A return implies that the mutex is now set for the caller. The condition must be re-checked before continuing. If the condition is still not true, the caller may wish to wait again.

`void Wait (i:PthreadInterval)`

Privilege: Public
No Inheritance
This operation initiates a wait on the condition variable. The caller should have set the mutex, then tested for the desired condition before making the call. A return implies that the mutex is now set for the caller. The condition must be re-checked before continuing. If the condition is still true, the caller may wish to wait again.

`void ~PthreadCond ()`

Privilege: Public
No Inheritance
The destructor deletes the Condition variable. It may only be called when there are no threads waiting on it, or on its mutex. No check is made to verify that this is the case.

5.2.2.53 Class PthreadInterval

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
PthreadCond (Aggregation)

Description:

The PthreadInterval class represents a time interval. This is used in several of the Pthread calls. Time intervals are distinguished from actual time (PthreadTime). Conversions may need to take place between time and interval, depending on the specific needs of a pthread intrinsic.

Attributes:

`tv_nsec`

Privilege: Private
Data Type: EcTLong
Default Value: NOT IDENTIFIED
No Inheritance
Additional nanoseconds since tv_sec

`tv_sec`

Privilege: Private
Data Type: EcTLong
Default Value: NOT IDENTIFIED
No Inheritance
Number of seconds since 00:00:00 GMT, 1 January 1970

Operations:

`EcTVoid PthreadInterval ()`

Privilege: Public
No Inheritance
The constructor builds a time interval.

`EcTVoid PthreadInterval (x:struct timespec)`

Privilege: Public
No Inheritance
The constructor builds a time interval.

`EcTVoid PthreadInterval (seconds:EcTLong)`

Privilege: Public
No Inheritance
The constructor builds a time interval.

`EcTVoid PthreadInterval (seconds:EcTLong nanosec:EcTLong)`

Privilege: Public
No Inheritance
The constructor builds a time interval.

5.2.2.54 Class PthreadMutex

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: Pthread(Public) can_be_controlled_by

Description:

The PthreadMutex class provides the fundamental locking mechanism.

Attributes:

Operations:

`void Lock ()`

Privilege: Public
No Inheritance
Acquires a lock on a mutex.

`void PthreadMutex ()`

Privilege: Public
No Inheritance
Construct a mutex.

`EcTInt TryLock ()`

Privilege: Public
No Inheritance
Attempts to acquire a lock on a mutex.

`void UnLock ()`

Privilege: Public
No Inheritance
Unlocks a mutex.

`void ~PthreadMutex ()`

Privilege: Public
No Inheritance
Delete the mutex. It is illegal to delete a mutex which is currently locked, but no checks are made.

5.2.2.55 Class PthreadTime

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
PthreadCond (Aggregation)

Description:

This class represents an actual time, as contrasted with a time interval represented by PthreadInterval. This class inherits timespec, as defined in pthread.h. The fields of this struct are publicly available. However, operations (including conversions) are preferable to direct use of timespec fields. The default copy constructor and assignment operator are available for use with this type.

Attributes:

`tv_nsec`

Privilege: Private
Data Type: EcTLong
Default Value: NOT IDENTIFIED
No Inheritance
Additional nanoseconds since tv_sec

`tv_sec`

Privilege: Private
Data Type: EcTLong
Default Value: NOT IDENTIFIED
No Inheritance
Number of seconds since 00:00:00 GMT, 1 January 1970

Operations:

`void PthreadTime ()`

Privilege: Public
No Inheritance
The constructor creates a PthreadTime. The timespec is initialized to the provided timespec, or to the current time plus indicated interval.

`void PthreadTime (x:struct timespec)`

Privilege: Public
No Inheritance
The constructor creates a PthreadTime. The timespec is initialized to the provided timespec, or to the current time plus indicated interval.

`void PthreadTime (a:PthreadInterval&)`

Privilege: Public
No Inheritance
The constructor creates a PthreadTime. The timespec is initialized to the provided timespec, or to the current time plus indicated interval.

5.2.2.56 Class ThisPthread

Synopsis:

Parent Class: Pthread
Is Not A Distributed Object
Is Associated With:
This class is derived from the class Pthread

Description:

ThisPthread is a reference to the current running thread. It is derived from Pthread. However, an object of type ThisPthread should never be referenced as a Pthread since the destructor has not been made virtual, and use of the Join and Stacksize member functions is disallowed.

Attributes:

`p`

Privilege: Private
Data Type: PthreadPrio
Default Value: Pthread_pri_mid
Inherited From: Pthread
Priority to run the thread(Pthread_pri_min, Pthread_pri_low, Pthread_pri_mid, Pthread_pri_high, Pthread_pri_max).

`param`

Privilege: Private
Data Type: ThreadParam
Default Value: NOT IDENTIFIED
Inherited From: Pthread
Parameter passed to thread when thread is started

`proc`

Privilege: Private
Data Type: ThreadProc
Default Value: NOT IDENTIFIED
Inherited From: Pthread
Procedure to be executed when thread is started

`s`

Privilege: Private
Data Type: PthreadSched
Default Value: Pthread_fg
Inherited From: Pthread
Scheduling type(Pthread_fifo- first in first out, Pthread_rr - round robin, Pthread_fg - foreground non portable, Pthread_bg - background non portable)

`size`

Privilege: Private
Data Type: EcTLong
Default Value: NOT IDENTIFIED
Inherited From: Pthread
Size of the stack for the thread

`t`

Privilege: Private
Data Type: PthreadTermination
Default Value: NOT IDENTIFIED
Inherited From: Pthread
Thread termination type. (Pthread_no_detach_on_delete, Pthread_detach_on_delete, or Pthread_join_on_delete)

Operations:

`void Delay (iv:PthreadInterval)`

Privilege: Public
No Inheritance
Delay this thread for a period of time.

`void Exit (result:PthreadResult)`

Privilege: Public
No Inheritance
Terminates execution of this thread, and sets the result that can be accessed by calling Join on this thread. Note: this function must not be called from the root thread.

`CancelState SetAsyncCancel (state:DCECancelState)`

Privilege: Public
No Inheritance
Control the cancellation states. CancelState is either CANCEL_OFF or CANCEL_ON.

`CancelState SetCancel (state:CancelState)`

Privilege: Public
No Inheritance
Control the cancellation states. CancelState is either CANCEL_OFF or CANCEL_ON

`void TestCancel ()`

Privilege: Public
No Inheritance
Tests whether cancellation of the thread has occurred.

`void ThisPthread ()`

Privilege: Public
No Inheritance
Constructor for the class

`void Yield ()`

Privilege: Public
No Inheritance
Allows another thread to gain control of kernel.

`void ~ThisPthread ()`

Privilege: Public
No Inheritance
Destructor for the class

`EcTVoid Cancel ()`

Privilege: Public
Inherited From: Pthread
Sends a cancel signal to the thread. This operation is only valid once a Pthread has been Start'ed. The thread may block cancellation, so it need not stop immediately.

`ThreadResult Join ()`

Privilege: Public
Inherited From: Pthread
Waits for completion of thread.

`void Priority ()`

Privilege: Public
Inherited From: Pthread
Retrieves priority

`PthreadPrio Priority (p:PthreadPrio)`

Privilege: Public
Inherited From: Pthread
Sets the priority of the thread

`void Pthread ()`

Privilege: Public
Inherited From: Pthread
Default constructor

`void Pthread (proc:ThreadProc param:ThreadParam)`

Privilege: Public
Inherited From: Pthread
Constructors for the Pthread class which will contruct a Pthread based on default attributes. The second form will also start the thread.

`void Scheduling ()`

Privilege: Public
Inherited From: Pthread
Retrieves scheduling algorithm

`PthreadSched Scheduling (s:PthreadSched p:PthreadPrio)`

Privilege: Public
Inherited From: Pthread
Sets scheduling and relative priority

`void Stacksize ()`

Privilege: Public
Inherited From: Pthread
Retrieves stack size

`EcTLong Stacksize (size:long)`

Privilege: Public
Inherited From: Pthread
Sets thread stack size

```
EcTVoid Start (proc:ThreadProc param:ThreadParam)
```

Privilege: Public
Inherited From: Pthread
Launches that thread by executing the specified procedure, passing it the specified parameter. This operation may only be performed on a Pthread that has no been previously Start'ed.

```
void Termination ()
```

Privilege: Public
Inherited From: Pthread
Retrieves termination options

```
void Termination (t:PthreadTermination)
```

Privilege: Public
Inherited From: Pthread
Sets termination options

```
void ~Pthread ()
```

Privilege: Public
Inherited From: Pthread
The destructor cleans up the Pthread data.

5.3 Data Dictionary Service Classes

5.3.1 Data Dictionary Service Classes Overview

This CSCI provides stores and provides access to descriptions of data products, their attributes, and the valid values of those attributes. Users query the DDICT to get these descriptions to enhance their knowledge of the system and what it provides. Terms can have different meanings based on the context they are used in. For example, the "Sea Surface Temperature" as a geophysical parameter can have a different meaning (units, location, etc.) dependent on which data product the parameter is used in. The DDICT provides these details to the user.

The DDICT data is also used by the other Data Management CSCIs to decompose queries and pass them on to other components. The attribute information is mapped to data products which are in turn mapped to components. Thus, the Distributed Information Manager and Local Information Manager can determine from the contents of a query which other components should be accessed to satisfy the request. The DDICT is the manager of this schema information.

The DDICT uses the Server Request Framework key mechanism documented in 305-CD-028-002 to provide both synchronous and asynchronous searching and schema request submission to the DDICT.

5.3.2 Data Dictionary Service Class Descriptions

5.3.2.1 Class DmDdRefrence

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DmDdResultSet (Aggregation)

Description:

From this object the calling object can get all the availbale information about a collection. the attribute list provides information for core and noncore attributes, including domain values and data types.

Attributes:

`myAttributeList`

Privilege: Private
Data Type: RWVector
Default Value: NOT IDENTIFIED
No Inheritance
the list of attributes available for search and access for this collection. This list will be a list of attribute classes from which the calling object can find out everything about an attribute.

`myCollectionDesc`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
The description of the data collection. This is a short summary description rather than the longer directory level description.

`myCollectionName`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
The name of the data collection.

`myGeoParameterList`

Privilege: Private
Data Type: RWVector
Default Value: NOT IDENTIFIED
No Inheritance

The list of geophysical parameters available in the data collection. This list will contain a parameter class which will provide all the information necessary about the geophysical parameters.

`myInstrument`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
The instrument that was used for the data collection. Data Type: RWVector

`myProviderList`

Privilege: Private
Data Type: RWVector
Default Value: NOT IDENTIFIED
No Inheritance
The list of providers (SDSRVs, LIMs, DIMS) that can search and access the data collection.

`mySatellite`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
The satellite that the instrument resided on.

Operations:

`RWVector GetAttrubteList (void)`

Privilege: Public
No Inheritance
Returns the list of attributes associated with the collection. This includes core as well as product specific attribtues.

`RWCString GetCollectionDesc (void)`

Privilege: Public
No Inheritance
Returns the list of myCollectionDesc attribute.

`void GetCollectionName (RWCString)`

Privilege: Public
No Inheritance
Returns the myCollectionName attribute.

`RWCString GetInstrument (void)`

Privilege: Public
No Inheritance
Returns myInstrument attribute.

`RWVector GetParameterList (void)`

Privilege: Public
No Inheritance
Returns the list go geophysical parameters that are contained in the data collection.

`RWVector GetProviderList (void)`

Privilege: Public
No Inheritance
Returns the list of provider that supply the collection. This provider list includes DIMGRs and SDSRVs.

`RWCString GetSatellite (void)`

Privilege: Public
No Inheritance
Returns the list of satellites available.

`void SetAttributeList (RWVector)`

Privilege: Public
No Inheritance
Sets te myAttributeList attribute to the value of the argument.

`void SetCollectionDesc (RWCString)`

Privilege: Public
No Inheritance
Sets the mycollectionDesc attribute to the value of the argument.

`void SetCollectionName (RWCString)`

Privilege: Public
No Inheritance
Sets the myCollectionName attribute to the value specified in the argument.

`void SetDiscipline (RWCString)`

Privilege: Public
No Inheritance
Set the mydiscipline attribute to the value of the argument.

`void SetGeoParameterList (RWVector)`

Privilege: Public
No Inheritance
Set the myGeoParameterList attribute to the value of the argument.

`void SetInstrument (RWCString)`

Privilege: Public
No Inheritance
Set the myInstrument attribute to the value of the argument.

`void SetProviderList (RWVector)`

Privilege: Public
No Inheritance

`void SetSatellite (RWCString)`

Privilege: Public
No Inheritance
Set the mySatellite attribute to the value of argument.

`void ~DmddRefrence_C ()`

Privilege: Public
No Inheritance

5.3.2.2 Class DmDdRequestServer_C

Synopsis:

Parent Class: EcCsRequestServer_C
Distributed Object
Is Associated With:
Class: EcCsMsg(Private) generates

Description:

This class is inheriting from EcCsRequestServer_C. It is used to manage the user session and to create objects capable of communicationg asynch between a client and a server.

Attributes:

`mySessonCommand`

Privilege: Private
Data Type: RWVector
Default Value: NOT IDENTIFIED
No Inheritance
Type of session command. Depending upon the command, NewSearchRequest () or NewSchemaRequest () method is called.

`myStatus`

Privilege: Private
Data Type: ECStatus
Default Value: NOT IDENTIFIED
No Inheritance
The status of the request.

`myUR`

Privilege: Private
Data Type: EcUrUR
Default Value: NOT IDENTIFIED
No Inheritance
The universal reference to the request on the server side.

`myUser`

Privilege: Private
Data Type: MSSUserProfile
Default Value: NOT IDENTIFIED
No Inheritance
This is the user profile of the user who created the request. It is used by the server to provide authorization to services and resources.

Operations:

`void DmDdRequestServer_C ()`

Privilege: Public
No Inheritance
Constructor to create an empty object.

```
void DmDdRequestServer_C (myUser:MSSUserProfile,
myServerUR:EcUrUR)
```

Privilege: Public
No Inheritance
Constructor called by a client application. User's profile is passed as a parameter.

`DmDdSchemaRequest_C* NewSchemaRequest (void)`

Privilege: Public
No Inheritance
This method is called by the client application to submit schema request. It returns a pointer to an asynchronous object DmDdSchemaRequest_C.

`DmDdSearchRequest* NewSearchRequest (void)`

Privilege: Public
No Inheritance
This method is called by the client application to submit search request..
It returns a pointer to an asynchronous obeject DmDdSearchRequest_C.

`RWBoolean SchemaDelete (DmDdSchemaMsg&)`

Privilege: Public
No Inheritance
This operation creates an object of class DmDdSchemaMsg.

`RWBoolean SchemaExport (DmDdSchemaMsg &)`

Privilege: Public
No Inheritance
This operation creates an object of class DmDdSchemaMsg.

`void ~DmDdrequestServer_C ()`

Privilege: Public
No Inheritance

5.3.2.3 Class DmDdResultSet

Synopsis:

Parent Class: RWPtrSlist
Is Not A Distributed Object
Is Associated With:
Class: DmDdSearchRequest_C(Public) creates

Description:

The object of this class is created by DmDdSearchRequest_C object to store set of results retrieved from the server.

Attributes:

`myPosition`

Privilege: Private
Data Type: EcTInt
Default Value: NOT IDENTIFIED
No Inheritance
The current position of the pointer in the result set.

`myUR`

Privilege: Private
Data Type: EcCsUrUR
Default Value: NOT IDENTIFIED
No Inheritance
The universal reference to this session.

Operations:

`void DmDdResultSet (void)`

Privilege: Public
No Inheritance
constructor to create an empty object.

`void DmDdResultSet (EcUrUR)`

Privilege: Public
No Inheritance

`DmDdRefrence& GetFirst (void)`

Privilege: Public
No Inheritance
Returns pointer to the first dictionary reference in the result set.

`DmDdRefrence& GetNext (void)`

Privilege: Public
No Inheritance
REturns a pointer to the next dictionary refernece in the result set.

`void ~DmDdResultSet ()`

Privilege: Public
No Inheritance
A destructor to destroy an object.

5.3.2.4 Class DmDdSchemaRequest_C

Synopsis:

Parent Class: EcCsAsynchRequest_C
Is Not A Distributed Object
Is Associated With:
This class is derived from the class EcCsAsynchRequest_C

Description:

This class is inheritng from EcCsAsynchRequest_C. It creates an asynchronous object for each schema request. The object is created by EcsMsgHandler. At the end of processing this object is notified by the EcsMsgHandler. Upon notification, this object creates DmDdResultSet object to get and store result.

Attributes:

`myRequestUR`

Privilege: Private
Data Type: EcUrUR
Default Value: NOT IDENTIFIED
No Inheritance
my server request UR.

`resultStatus`

Privilege: Private
Data Type: RWBoolean
Default Value: NOT IDENTIFIED
No Inheritance
Status of the result. Set to TRUE when result is received.

Operations:

`void DmDdSchemaRequest_C ()`

Privilege: Public
No Inheritance

`RWBoolean GetResultSet (void)`

Privilege: Public
No Inheritance
This method creates DmDdResultSet object to retrieve and store results.

`RWBoolean Resume (void)`

Privilege: Public
No Inheritance
This method resumes request submitted for a schema search to the DDICT server.

`void SubmitSearch (void)`

Privilege: Public
No Inheritance
This method suspends request submitted for a schema search to the DDICT server.

`RWBoolean SuspendSearch (void)`

Privilege: Public
No Inheritance
This method suspends request submitted for a schema search to the DDICT server.

`void ~DmDdSchemaRequest_C ()`

Privilege: Public
No Inheritance

5.3.2.5 Class DmDdSearchRequest_C

Synopsis:

Parent Class: EcCsAsynchRequest_C
Is Not A Distributed Object
Is Associated With:
Class: DmDdResultSet(Public) creates

Description:

This class is inheriting from EcCsAsynchRequest_C. this object is constructed via callback by the EcCsMsgHandler. It is notified by the EcCsMsgHandler about the status of the request at the server end as well as at the end of processing. Upon completion of processing, this object creates DmDdResultSet object to store result.

Attributes:

`myCallback`

Privilege: Private
Data Type: DmImCallback
Default Value: NOT IDENTIFIED
No Inheritance
The callback to notify an object about the result status.

`myRequestUR`

Privilege: Private
Data Type: EcUrUR
Default Value: NOT IDENTIFIED
No Inheritance
my server request UR.

`resultStatus`

Privilege: Private
Data Type: RWBoolean
Default Value: NOT IDENTIFIED
No Inheritance
Status of the result. Set to TRUE when result is received. .

Operations:

`void DmDdSearchRequest_C ()`

Privilege: Public
No Inheritance
A constructor called by DmDdRequestServer_C object to create an asych object.

`DmDdResultSet* GetResultSet (void)`

Privilege: Public
No Inheritance
this method creates DmDdResultSet object to store results received from the DDICT server.

`RWBoolean GetResultStatus (void)`

Privilege: Public
No Inheritance
Status of the result. Set to TRUE when result is received.

`RWBoolean Resume (void)`

Privilege: Public
No Inheritance
This method resumes request submitted for a schema search to the DDICT server.

`void StatChange (void)`

Privilege: Public
No Inheritance
This operation indicates change of status of the result received.

`EcUrUR SubmitSearch (RWCString)`

Privilege: Public
No Inheritance

`RWBoolean SuspendSearch (void)`

Privilege: Public
No Inheritance
This method suspends request submitted to the server.

`void setCallback (DmImCallback*)`

Privilege: Public
No Inheritance
sets myCallback attribute.

```
void ~DmDdSearchRequest_C ()
```

Privilege: Public
No Inheritance

5.4 Data Distribution Classes

5.4.1 Data Distribution Classes Overview

Data Distribution CSCI: The separation between the "push" and "pull" sides of the system into provider and consumer domains is essential for the data server to meet its requirements for highly available and reliable archival and distribution of data. The Data Distribution CSCI manages and provides access to the resources in the data "pull" side of operations. This facilitates the scaling and customization of distribution hardware, software, and operations for each operational site, while isolating the changes required to access and manage site specific requirements to this CSCI.

5.4.2 Data Distribution Class Descriptions

5.4.2.1 Class DsDdDataItem

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: DsDdMedia(Private) iswrittento
DsDdGranuleB (Aggregation)

Description:

Base class for items to be distributed. For now only a file specialization of this class exists and is used, but the base class is defined to support possible future use of streams et al.

Attributes:

```
myCompressedSizeB
```

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
Size of a data item when compressed, in megabytes.

```
myCompressionTypeB
```

Privilege: Private
Data Type: RWCString
Default Value: "none"
No Inheritance
Type of compression, if any, performed on the item. Compression types are TBD.

`myID`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
Identifier, such as filename, for the data item.

`myUncompressedSizeB`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
Size of the object before it was compressed.

`myWhetherCompressedB`

Privilege: Private
Data Type: RWBoolean
Default Value: false
No Inheritance
Flag indicating whether or not this data item is compressed.

Operations:

`EcUtStatus CompressB (Type: RWCString *)`

Privilege: Public
No Inheritance
Method to compress the data item with provided compression type.

5.4.2.2 Class DsDdDistFile

Synopsis:

Parent Class: DsDdDataItem
Is Not A Distributed Object
Is Associated With:
Class: DsStStagingDisk(Private) residesupon - Each file to be distributed will be retrieved (via STMGT) to staging disk.

Description:

File containing data to be distributed to the requestor.

Attributes:

`myPath`

Privilege: Private
Data Type: RWCString *
Default Value: NOT IDENTIFIED
No Inheritance
Specification of where the file is located, including device and directory.

`myRetrievedFlagB`

Privilege: Private
Data Type: RWBoolean
Default Value: false
No Inheritance
Indicates whether the file has been retrieved from archive. Some files - such as subsetted files - are retrieved by SDSRV prior to the request being passed to DDIST. Those which aren't retrieved by SDSRV - such as any file that does not go through user-requested processing prior to DDIST getting passed the request - must be retrieved by DDIST.

`myCompressedSizeB`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: DsDdDataItem
Size of a data item when compressed, in megabytes.

`myCompressionTypeB`

Privilege: Private
Data Type: RWCString
Default Value: "none"
Inherited From: DsDdDataItem
Type of compression, if any, performed on the item. Compression types are TBD.

`myID`

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
Inherited From: DsDdDataItem
Identifier, such as filename, for the data item.

`myUncompressedSizeB`

Privilege: Private
Data Type: EcTInt
Default Value: 0
Inherited From: DsDdDataItem
Size of the object before it was compressed.

`myWhetherCompressedB`

Privilege: Private
Data Type: RWBoolean
Default Value: false
Inherited From: DsDdDataItem
Flag indicating whether or not this data item is compressed.

Operations:

`EcUtStatus CompressB ()`

Privilege: Public
No Inheritance
Compress the file. At a minimum Unix file compression will be supported.

`DistFile Flatten ()`

Privilege: Public
No Inheritance
Express the object as a byte stream so it can be transported from the client to the server via OODCE.

`EcUtStatus CompressB (Type: RWCString *)`

Privilege: Public
Inherited From: DsDdDataItem
Method to compress the data item with provided compression type.

5.4.2.3 Class DsDdDistList

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: DsSrCost(Private) calculates
Class: DsDdDistRequest(Public) hasa
Class: DsDdRequestManager(Public) receives
Class: DsDdDistRequest(Public) requestsdistributionofitemsin - A distribution request is the vehicle for performing the distribution of the items in the distribution list.

Description:

List of pointers to all the items to be distributed. This class is derived from a RogueWave template, and so inherits all of the operations (not shown) to manipulate the list.

Attributes:

`myCompressionTypeB`

Privilege: Private
Data Type: RWCString
Default Value: "None"
No Inheritance
Type of compression - e.g., Unix compression - to be applied on the entire list.

`myDistSize`

Privilege: Private
Data Type: EcTInt
Default Value: 0
No Inheritance
Size, in megabytes of the entire list of iles to be distributed.

Operations:

`EcUtStatus CompressB ()`

Privilege: Public
No Inheritance
Compress the data included in this list, using the compression specified by myCompressionTypeB.

`DsDdDistRequest Distribute (Media: RWCString *, User: MsUsProfile, Format: RWCString *)`

Privilege: Public
No Inheritance
Initiates distribution processing for all of the items in the list. This signature definition supports all requests except for electronic push requests.

`Distribute (Media: RWCString *, User: MsUsProfile, ElecDestn: RWCString *, Password: RWCStri ng *, Format: RWCString *)`

Privilege: Protection Not Identified
No Inheritance

`DsUzCost Estimate ()`

Privilege: Private
No Inheritance
Estimate the time required and the cost of distributing the items included in this list.

`DistList Flatten ()`

Privilege: Public
No Inheritance
Express the object's attributes as a byte stream. This is necessary because this object is referenced by the distributed object DsDdDistRequest. Because OODCE does not support deep copy (of referenced objects) from client- to server-side, referenced objects must be expressed as a byte stream before they are transferred to the server side.

`EcUtStatus RetrieveB ()`

Privilege: Public
No Inheritance
Retrieve from the archive any data which needs to be retrieved before being distributed. Some of the data may already have been retrieved by the SDSRV CSCI in preparation for this distribution request; for example, some data may have been retrieved for subsetting or other manipulation prior to distribution.

`void SumTheSize (EcTInt)`

Privilege: Private
No Inheritance
Calculate the total size of the data to be distributed.

5.4.2.4 Class DsDdDistRequest

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
Class: DsDdRequestManager(Public) createsmanages
Class: DsDdDistList(Public) hasa
Class: DsDdRequestProcessor(Private) isexecutedby
Class: DsDdDistList(Public) requestsdistributionofitemsin - A distribution request is the vehicle for performing the distribution of the items in the distribution list.
DsDdRequestLIst (Aggregation)

Description:

Base class for the client/server specialization of distribution requests. Specifies operations which are available to any client.

Attributes:

Operations:

`EcUtStatus Abort ()`

Privilege: Public
No Inheritance
Terminate all processing of a distribution request. The termination may not occur immediately, because the request may need to progress to a stable stage enabling clean termination.

`EcStatus CreateSubrequestB ()`

Privilege: Private
No Inheritance
Create a new request from a portion of an existing request/list of files to distribute. This is used in conjunction with the DelimitB method to process requests which are too large - in bytes or number of files - to process as a single request.

`EcTInt DelimitB ()`

Privilege: Public
No Inheritance
Indicate where in the list of granules/files to be distributed to break the request into smaller requests. This is necessary for requests that are too large - in terms of bytes or number of files - to be processed as a single request.

`DsUzCost EstimateB ()`

Privilege: Public
No Inheritance

`DistState GetState ()`

Privilege: Public
No Inheritance
Get the state of the distribution request. States defined in the L4 requirements are: pending, active, waiting for shipment, shipped.

```
void Submit (Media: RWCString *, User: MsUsProfile, Format:
RWCString *)
```

Privilege: Private
No Inheritance
Submit a distribution request for processing. This definition of the submit signature supports all distribution except electronic push, which requires additional arguments.

```
void Submit ((Media: RWCString *, User: MsUsProfile,
ElecDestn: RWCString *, Password: RWCString *, Format:
RWCString *)
```

Privilege: Private
No Inheritance
Submit a distribution request for processing. This definition of the submit signature supports electronic push only.

```
EcUtStatus WaitForComplete ()
```

Privilege: Public
No Inheritance
Wait until a distribution request has completed. Since the submission of a distribution request is asynchronous, this service allows a thread to block until (and thereby be signalled when) the requests completes.

5.4.2.5 Class DsDdDistRequestC

Synopsis:

Parent Class: DsDdDistRequest
Distributed Object
Is Associated With:
This class is derived from the class DsDdDistRequest

Description:

User client presentation of the distribution request. A specialization of the constructor is necessary to accomodate hiding of the request manager (which implements the OODCE factory model) from the client.

Attributes:

Operations:

```
DCEObjRefT DsDdDistRequestC ()
```

Privilege: Public
No Inheritance
Constructor for the user client presentation of the distribution request. This constructor implements the client side of the OODCE factory model; it first creates a client factory object if it doesn't exist, then calls the appropriate factory service to create a server-side distribution request.

`EcUtStatus Abort ()`

Privilege: Public
Inherited From: DsDdDistRequest
Terminate all processing of a distribution request. The termination may not occur immediately, because the request may need to progress to a stable stage enabling clean termination.

`EcStatus CreateSubrequestB ()`

Privilege: Private
Inherited From: DsDdDistRequest
Create a new request from a portion of an existing request/list of files to distribute. This is used in conjunction with the DelimitB method to process requests which are too large - in bytes or number of files - to process as a single request.

`EcTInt DelimitB ()`

Privilege: Public
Inherited From: DsDdDistRequest
Indicate where in the list of granules/files to be distributed to break the request into smaller requests. This is necessary for requests that are too large - in terms of bytes or number of files - to be processed as a single request.

`DsUzCost EstimateB ()`

Privilege: Public
Inherited From: DsDdDistRequest

`DistState GetState ()`

Privilege: Public
Inherited From: DsDdDistRequest
Get the state of the distribution request. States defined in the L4 requirements are: pending, active, waiting for shipment, shipped.

`void Submit (Media: RWCString *, User: MsUsProfile, Format: RWCString *)`

Privilege: Private
Inherited From: DsDdDistRequest
Submit a distribution request for processing. This definition of the submit signature supports all distribution except electronic push, which requires additional arguments.

```
void Submit ((Media: RWCString *, User: MsUsProfile,
ElecDestn: RWCString *, Password: RWCString *, Format:
RWCString *)
```

Privilege: Private
Inherited From: DsDdDistRequest
Submit a distribution request for processing. This definition of the submit signature supports electronic push only.

```
EcUtStatus WaitForComplete ()
```

Privilege: Public
Inherited From: DsDdDistRequest
Wait until a distribution request has completed. Since the submission of a distribution request is asynchronous, this service allows a thread to block until (and thereby be signalled when) the requests completes.

5.4.2.6 Class DsDdGranuleB

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DsDdDistList (Aggregation)

Description:

This class is a collection of all of the files which compose a granule to be distributed.

Attributes:

```
myTypeofConversion
```

Privilege: Private
Data Type: RWCString
Default Value: NOT IDENTIFIED
No Inheritance
The format to which the granule is to be converted. The finest granularity at which conversion can be specified is the granule.

Operations:

```
EcUtStatus Convert ()
```

Privilege: Public
No Inheritance
Convert the granule to the format specified by myTypeofConversion.

5.4.2.7 Class DsDdOpsInterventionListB

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
None

Description:

Container of Distribution requests requiring operator intervention. This class is a RogueWave RWOrdered class.

Attributes:

Operations:

5.4.2.8 Class DsDdOpsRequestC

Synopsis:

Parent Class: DsDdPrivRequest
Distributed Object
Is Associated With:
This class is derived from the class DsDdPrivRequest

Description:

Presentation of the distribution request services to the operator client. This client has access to privileged services which are not available to the user client.

Attributes:

Operations:

```
DCEObjRefT DsDdOpsRequestC ()
```

Privilege: Public
No Inheritance
The constructor for the operator client request interfaces to the request factory (manager) for creation of the server-side request.

```
EcUtStatus RestartOutput ()
```

Privilege: Public
Inherited From: DsDdPrivRequest
Restart the output to media of a distribution request. This service is provide to support the operator restarting media output in the event of a failure, such as fatal tape errors during writing of a tape.

`void SetPriority (Priority:DistPriority)`

Privilege: Public
Inherited From: DsDdPrivRequest
Changes the priority of a distribution request, which the operator may need to do when managing the flow of distribution requests.

`EcUtStatus Abort ()`

Privilege: Public
Inherited From: DsDdDistRequest
Terminate all processing of a distribution request. The termination may not occur immediately, because the request may need to progress to a stable stage enabling clean termination.

`EcStatus CreateSubrequestB ()`

Privilege: Private
Inherited From: DsDdDistRequest
Create a new request from a portion of an existing request/list of files to distribute. This is used in conjunction with the DelimitB method to process requests which are too large - in bytes or number of files - to process as a single request.

`EcTInt DelimitB ()`

Privilege: Public
Inherited From: DsDdDistRequest
Indicate where in the list of granules/files to be distributed to break the request into smaller requests. This is necessary for requests that are too large - in terms of bytes or number of files - to be processed as a single request.

`DsUzCost EstimateB ()`

Privilege: Public
Inherited From: DsDdDistRequest

`DistState GetState ()`

Privilege: Public
Inherited From: DsDdDistRequest
Get the state of the distribution request. States defined in the L4 requirements are: pending, active, waiting for shipment, shipped.

```
void Submit (Media: RWCString *, User: MsUsProfile, Format:
RWCString *)
```

Privilege: Private
Inherited From: DsDdDistRequest
Submit a distribution request for processing. This definition of the submit signature supports all distribution except electronic push, which requires additional arguments.

```
void Submit ((Media: RWCString *, User: MsUsProfile,
ElecDestn: RWCString *, Password: RWCString *, Format:
RWCString *)
```

Privilege: Private
Inherited From: DsDdDistRequest
Submit a distribution request for processing. This definition of the submit signature supports electronic push only.

```
EcUtStatus WaitForComplete ()
```

Privilege: Public
Inherited From: DsDdDistRequest
Wait until a distribution request has completed. Since the submission of a distribution request is asynchronous, this service allows a thread to block until (and thereby be signalled when) the requests completes.

5.4.2.9 Class DsDdPrivRequest

Synopsis:

Parent Class: DsDdDistRequest
Is Not A Distributed Object
Is Associated With:
This class is derived from the class DsDdDistRequest

Description:

Specialization of the distribution request to provide access to privileged operations. Privileged operations are available to operator clients but not user clients.

Attributes:

Operations:

`EcUtStatus RestartOutput ()`

Privilege: Public
No Inheritance
Restart the output to media of a distribution request. This service is provide to support the operator restarting media output in the event of a failure, such as fatal tape errors during writing of a tape.

`void SetPriority (Priority:DistPriority)`

Privilege: Public
No Inheritance
Changes the priority of a distribution request, which the operator may need to do when managing the flow of distribution requests.

`EcUtStatus Abort ()`

Privilege: Public
Inherited From: DsDdDistRequest
Terminate all processing of a distribution request. The termination may not occur immediately, because the request may need to progress to a stable stage enabling clean termination.

`EcStatus CreateSubrequestB ()`

Privilege: Private
Inherited From: DsDdDistRequest
Create a new request from a portion of an existing request/list of files to distribute. This is used in conjunction with the DelimitB method to process requests which are too large - in bytes or number of files - to process as a single request.

`EcTInt DelimitB ()`

Privilege: Public
Inherited From: DsDdDistRequest
Indicate where in the list of granules/files to be distributed to break the request into smaller requests. This is necessary for requests that are too large - in terms of bytes or number of files - to be processed as a single request.

`DsUzCost EstimateB ()`

Privilege: Public
Inherited From: DsDdDistRequest

```
DistState GetState ()
```

Privilege: Public
Inherited From: DsDdDistRequest
Get the state of the distribution request. States defined in the L4 requirements are: pending, active, waiting for shipment, shipped.

```
void Submit (Media: RWCString *, User: MsUsProfile, Format:
RWCString *)
```

Privilege: Private
Inherited From: DsDdDistRequest
Submit a distribution request for processing. This definition of the submit signature supports all distribution except electronic push, which requires additional arguments.

```
void Submit ((Media: RWCString *, User: MsUsProfile,
ElecDestn: RWCString *, Password: RWCString *, Format:
RWCString *)
```

Privilege: Private
Inherited From: DsDdDistRequest
Submit a distribution request for processing. This definition of the submit signature supports electronic push only.

```
EcUtStatus WaitForComplete ()
```

Privilege: Public
Inherited From: DsDdDistRequest
Wait until a distribution request has completed. Since the submission of a distribution request is asynchronous, this service allows a thread to block until (and thereby be signalled when) the requests completes.

5.4.2.10 Class DsDdRequestList

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DsDdOpsInterventionListB (Aggregation)

Description:

Set of pointers to all distribution requests.

Attributes:

Operations:

`DsDdRequestList Sort (Key:SortTypes)`

Privilege: Public
No Inheritance
Sort the requests by a particular attribute. Level 4 requirements exist to display requests by request id, state, or request type (electronic or physical media).

5.4.2.11 Class DsDdRequestManager

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
Class: DsDdDistRequest(Public) createsmanages
Class: DsDdDistList(Public) receives
Class: DsDdRequestProcessor(Private) submitsrequeststo
Class: DsDdRequestLIst(Private) worksrequestsoffof

Description:

Base class for implementation of the OODCE factory model. Derived classes manufacture server-side distribution requests and provide client-side presentations of those requests.

Attributes:

Operations:

```
DCEObjRefT CreateDistRequest (List: DistList, Media:
RWCString *, User: MsUsProfile, Format: RWCString *)
```

Privilege: Public
No Inheritance
Creates a new distribution request. Two definitions of this service exists; this definition creates a distribution request for all requests except for electronic push requests, which require additional arguments.

```
CreateDistRequest (List: DistList, Media: RWCString *, User:
MsUsProfile, ElecDestntn: RWCString *, Password: RWCString *,
Format: RWCString *)
```

Privilege: Protection Not Identified
No Inheritance

```
DCEObjRefT CreateDistSubRequestB (list: DsDdDistList, media:
RWCString, user: MsUsUserProfile, format: RWCString)
```

Privilege: Public
No Inheritance
Create Subrequests if a distribution request is too large.

```
DCEObjRefT CreateDistSubRequestB (list: DsDdDistList, media:
RWCString, user: MsUsUserProfile, dest: RWCString, pword:
RWCString, format: RWCString)
```

Privilege: Public
No Inheritance
Overload member for electronic distribution that creates subrequests if a distribution request is too large.

```
void DeleteDistRequest (id: uuid_t)
```

Privilege: Public
No Inheritance
Service, in the OODCE factory model, to delete a distribution request which was created via the CreateDistRequest service.

```
EcUtStatus DeleteDistSubRequestB (id: uuid_t)
```

Privilege: Public
No Inheritance
Delete a subrequest.

```
DsDdRequestList InventoryRequests ()
```

Privilege: Public
No Inheritance
Provides an inventory - in the form of a list of distributed object references - of all distribution requests.

5.4.2.12 Class DsDdRequestManagerC

Synopsis:

Parent Class: DsDdRequestManager
Distributed Object
Is Associated With:
This class is derived from the class DsDdRequestManager

Description:

Client-side presentation of the request factory. The client's applications code will be unaware of the existence of this object, because this object will be created and managed from within the client instance of the distribution request. While this may seem to be a chicken and egg paradox - the distribution request creates the factory, which creates the distribution request - is isn't, because the logic is actually: 1- the client-side distribution request is created, which 2- creates the factory, which 3- is used to create a server-side instance of the distribution request, 4- whose OODCE distributed reference is returned, to the client caller, as the client-side distribution request

Attributes:

Operations:

`DCEObjRefT CreateDistRequest (List: DistList, Media: RWCString *, User: MsUsProfile, Format: RWCString *)`

Privilege: Public
Inherited From: DsDdRequestManager
Creates a new distribution request. Two definitions of this service exists; this definition creates a distribution request for all requests except for electronic push requests, which require additional arguments.

`CreateDistRequest (List: DistList, Media: RWCString *, User: MsUsProfile, ElecDestntn: RWCString *, Password: RWCString *, Format: RWCString *)`

Privilege: Protection Not Identified
Inherited From: DsDdRequestManager

`DCEObjRefT CreateDistSubRequestB (list: DsDdDistList, media: RWCString, user: MsUsUserProfile, format: RWCString)`

Privilege: Public
Inherited From: DsDdRequestManager
Create Subrequests if a distribution request is too large.

`DCEObjRefT CreateDistSubRequestB (list: DsDdDistList, media: RWCString, user: MsUsUserProfile, dest: RWCString, pword: RWCString, format: RWCString)`

Privilege: Public
Inherited From: DsDdRequestManager
Overload member for electronic distribution that creates subrequests if a distribution request is too large.

`void DeleteDistRequest (id: uuid_t)`

Privilege: Public
Inherited From: DsDdRequestManager
Service, in the OODCE factory model, to delete a distribution request which was created via the CreateDistRequest service.

`EcUtStatus DeleteDistSubRequestB (id: uuid_t)`

Privilege: Public
Inherited From: DsDdRequestManager
Delete a subrequest.

`DsDdRequestList InventoryRequests ()`

Privilege: Public
Inherited From: DsDdRequestManager
Provides an inventory - in the form of a list of distributed object references - of all distribution requests.

5.5 Document Data Server Classes

5.5.1 Document Data Server Classes Overview

The allocation of document search and retrieval to the Document Data Server CSCI enhances the ability of the data server to respond to emerging technologies in this area. It is anticipated that the Document Data Server CSCI will consist of primarily COTS software components, and will provide World-Wide-Web style browsing and searching of document data types.

5.5.2 Document Data Server Class Descriptions

5.5.2.1 Class DsCtAcquireCommand

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
DsCtCommand (Aggregation)

Description:

This object represents the document retrieval commands received for the document data.

Attributes:

`$myDsCtAcquireCommand`

Privilege: Private
Data Type: List <DsCtAcquireCommand>
Default Value: null
No Inheritance
List of all active acquire commands. Used mainly for fault recovery and memory management.

`myCommand`

Privilege: Private
Data Type: DsCtCommand &
Default Value: null
No Inheritance
Reference to the associated command.

`myDsEsESDT`

Privilege: Private
Data Type: DsEsESDT &
Default Value: null
No Inheritance
Reference to the document ESDT associated with the acquire command. The CSDT related to this ESDT is used to implement the extract operation.

`myHTTPRequest`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
HTTP Get command string. Used to identify the location of the document to return.

`myOstr`

Privilege: Private
Data Type: ostream &
Default Value: null
No Inheritance
Output stream to write document data.

Operations:

`GlStatus AcquireCommand ()`

Privilege: Public
No Inheritance
This is the actual acquire command received for the Document data.

`void DsCtAcquireCommand (DsCtCommand &)`

Privilege: Public
No Inheritance
This object represents the constructor for the commands received for acquiring the Document Data.

`void ~DsCtAcquireCommand ()`

Privilege: Public
No Inheritance
Object destructor - invoke ESDT destructor.

5.5.2.2 Class DsCtClient

Synopsis:

Parent Class: DsDoClient
Distributed Object
Is Associated With:
Class: DsCtRequest(Public)
Class: DsCtRequest(Public) constructs
Class: DsSvServer(Public) manages
Class: DsSvServer(Public) serves

Description:

This object is the client object for the server object of Document Data server.

Attributes:

`$myClientList`

Privilege: Private
Data Type: List <DsCtCleint>
Default Value: null
No Inheritance
List of active clients.

`myHostAddress`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Internet address of client's host.

`myIPnumber`

Privilege: Private
Data Type: int
Default Value: 0
No Inheritance
IP address of the client's host.

`myPortNumber`

Privilege: Private
Data Type: int
Default Value: 0
No Inheritance
TCP/IP port number of the client.

`myProtocolName`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the protocol used for the client/server connection. For external connections the protocol will be HTTP V1.0

`myProtocolVersion`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Version of the communication protocol used by the client. For external connections the protocol will be HTTP V1.0

`mySecurityProtocol`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the security protocol used by the client.

`mySecurityProtocolVersion`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Version of the client's security protocol.

`myServer`

Privilege: Private
Data Type: DsSvServer &
Default Value: null
No Inheritance
Reference to the clients associated server.

`myClientName`

Privilege: Private
Data Type: char *
Default Value: null
Inherited From: DsDoClient
Name of the client. Used to identify the protocol and behaviour of a WWW client.

`myClientVersion`

Privilege: Private
Data Type: char *
Default Value: null
Inherited From: DsDoClient
Version number of the client. Different WWW clients may have differnt behaviour and different versions may present a different interface.

`mySystemLog`

Privilege: Private
Data Type: GlLog &
Default Value: null
Inherited From: DsDoClient
Reference to the system log for exception reporting and logging.

Operations:

`GlStatus ConnectServer (DsSvServer)`

Privilege: Private
No Inheritance
Make a connection to the server. This connection is synchronous and is established for the duration of the transaction to service the request.

`GlStatus DisConnectServer ()`

Privilege: Private
No Inheritance
Disconnection of the client from the server. This operation is invoked when the external client unexpectedly quits. For example the user may select the stop button on the WWW Client interface, this breaks the TCP/IP socket and the server is notified. All outstanding service requests for the client need to be aborted.

`void DsCtClient (MSS_UserProfile)`

Privilege: Public
No Inheritance
Object constructor - for internal ECS connections the MSS_UserProfile must be specified. For external connections via HTTP, a default profile will be used which allows read access only.

`GlStatus SubmitRequest (DsCtRequest)`

Privilege: Public
No Inheritance
Submit a request to be executed by the server. The DsCtCommand and DsCtRequest structures should be constructed before this operation is invoked. The parameters described in each command should be validated prior before it is submitted to the server.

`void ~DsCtClient ()`

Privilege: Public
No Inheritance
Object destructor - no specific implementation

5.5.2.3 Class DsCtCommand

Synopsis:

Parent Class: DsDoCommand
Distributed Object
Is Associated With:
DsCtRequest (Aggregation)

Description:

This object represents the client commands received for Document data.

Attributes:

`$myCommandList`

Privilege: Private
Data Type: List <DsCtCommand>
Default Value: null
No Inheritance
List of all active commands.

`myCommandName`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the service to execute.

`myCommandRequest`

Privilege: Private
Data Type: DsCtRequest &
Default Value: null
No Inheritance
Associated request.

`myCommandType`

Privilege: Private
Data Type: enum
Default Value: 0
No Inheritance
Command type to execute.

`myCommandType`

Privilege: Private
Data Type: enum {}
Default Value: 0
Inherited From: DsDoCommand
Type of command to execute. Relates to the service request type.

`myRequest`

Privilege: Private
Data Type: DsDoRequest &
Default Value: null
Inherited From: DsDoCommand
Reference to the associated request.

Operations:

`void DsCtCommand (char * FileName)`

Privilege: Public
No Inheritance
Constructor to read command from specified file.

`void DsCtCommand (istream *istr)`

Privilege: Public
No Inheritance
Input stream for DsCt Command class.

`GlStatus ProcessCommand (int)`

Privilege: Public
No Inheritance
Called to execute this command. The appropriate sub-command operation is invoked.

`void ~DsCtCommand ()`

Privilege: Public
No Inheritance
Object destructor - no specific implementation.

`void ExecuteCommand ()`

Privilege: Public
Inherited From: DsDoCommand
Called to invoke the appropriate service object according to the command type.

5.5.2.4 Class DsCtInsertCommand

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
DsCtCommand (Aggregation)

Description:

This object represents the insert commands received for the document data.

Attributes:

`$DsCtInsertCommandList`

Privilege: Private
Data Type: List <DsCtInsertCommand>
Default Value: null
No Inheritance
Lists of DsCtInsertCommands received for document data.

`myCommand`

Privilege: Private
Data Type: DsCtCommand &
Default Value: null
No Inheritance
Associated DsCtCommand Object.

`myDatafile`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Document data to be inserted in the document repository.

`myDsEsESDT`

Privilege: Private
Data Type: DsEsESDT &
Default Value: null
No Inheritance
Reference to ESDT for internalize operation.

`myMetaFile`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Associated PVL file containing the metadata to be submitted to the DBMS wrapper layer.

Operations:

`void DsCtInsertCommand (DsCtCommand &)`

Privilege: Public
No Inheritance
Constructor with reference to assocaited command.

`GlStatus InsertCommand ()`

Privilege: Public
No Inheritance
Invoked for the insertion of a document into the document repository. The associated DsEsESDT object is called with an internalize() method to insert the metadata into the DBMS through the DBMS wrapper layer. Returns a GlStatus to indicate success or failure.

5.5.2.5 Class DsCtRequest

Synopsis:

Parent Class: DsDoRequest
Distributed Object
Is Associated With:
Class: DsCtClient(Public)
Class: DsSvServer(Public)
Class: DsCtClient(Public) constructs
Class: DsSvServer(Public) services

Description:

This object represents the requests from client to the Document Data Server.

Attributes:

`$myRequestList`

Privilege: Private
Data Type: List <DsCtRequest>
Default Value: null
No Inheritance
List of currently active requests. Used for fault recovery and memory management.

`myCommndList`

Privilege: Private
Data Type: List <DsCtCommand>
Default Value: null
No Inheritance
List of the commands associated with this request.

`myRequestStartTime`

Privilege: Private
Data Type: RWDateTime &
Default Value: null
No Inheritance
Date and time of request submition.

`myRequestStatus`

Privilege: Private
Data Type: GlStatus &
Default Value: null
No Inheritance
Current status of request.

`myRequestTimeOut`

Privilege: Private
Data Type: RWDateTime &
Default Value: null
No Inheritance
Date and Time for request to time out.

`myClient`

Privilege: Private
Data Type: DsDoClient &
Default Value: null
Inherited From: DsDoRequest
Reference to the associated client for this request.

`myRequestName`

Privilege: Private
Data Type: char *
Default Value: null
Inherited From: DsDoRequest
Name of this request - taken from the GlParameterList for this request.

`myRequestType`

Privilege: Private
Data Type: int
Default Value: null
Inherited From: DsDoRequest
Type of request to be serviced.

`myServer`

Privilege: Private
Data Type: DsDoServer &
Default Value: null
Inherited From: DsDoRequest
Reference to the associated server for this request.

Operations:

`GlStatus CancelRequest ()`

Privilege: Public
No Inheritance
Called to cancel a currently active request.

`void DsCtRequest ()`

Privilege: Public
No Inheritance
Default constructor.

`GlStatus RequestStaus ()`

Privilege: Public
No Inheritance
Return the status of the currently executing request.

`GlStatus ServiceRequest ()`

Privilege: Public
No Inheritance
Called to execute the commands associated with this request. Loop through the command list and execute the appropriate service request.

`void ~DsCtRequest ()`

Privilege: Public
No Inheritance
Object destructor - call destructors for associated commands.

`void DsCIRequest ()`

Privilege: Public
Inherited From: DsDoRequest
Object constructor - no specific implementation

`GlStatus ServiceRequest ()`

Privilege: Public
Inherited From: DsDoRequest
Invoked for the execution of this service request, normally from the server after the client has constructed the request and called service request to the server.

`void ~DsCIRequest ()`

Privilege: Public
Inherited From: DsDoRequest
Object destructor - call the destructor of the associated commands.

5.5.2.6 Class DsCtSearchcommand

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
DsCtCommand (Aggregation)

Description:

This object represents the search commands received for the document data.

Attributes:

`$myDsCtSearchCommandList`

Privilege: Private
Data Type: List <DsCtSearchCommand>
Default Value: null
No Inheritance
Lists all the search commands received for the document data from the client.

`myCommand`

Privilege: Private
Data Type: DsCtCommand &
Default Value: null
No Inheritance
Reference to associated command object.

`myFreeTextResultsList`

Privilege: Private
Data Type: List <char *>
Default Value: null
No Inheritance
The list of URLs returned as a result of a free text query submitted to the COTS search engine.

`myHTMLResultsList`

Privilege: Private
Data Type: List <char *>
Default Value: null
No Inheritance
The list of URLs to be packaged in a HTML document which is returned to the WWW client across the HTTP connection.

`myKeywordResultsList`

Privilege: Private
Data Type: List <char *>
Default Value: null
No Inheritance
The list of URLs returned as a result of a keyword search submitted to the DBMS wrapper layer.

`myParameterList`

Privilege: Private
Data Type: GlParameterList *
Default Value: null
No Inheritance
The GlParameterList which represents the query to be submitted to the DBMS wrapper layer. These search parameters are extracted from the WAIS query string.

`myResultsList`

Privilege: Private
Data Type: GlParameterList *
Default Value: null
No Inheritance
The results set returned from the DBMS wrapper layer. From this list the matching URLs are copied to the keyword results list.

`myWAISQuery`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
WAIS query to be executed. The query string is copied from the QUERY_STRING environmental passed through the CGI interface call. The query may be keyword or free text.

Operations:

`void DsCSearchCommand (DsCtCommand &)`

Privilege: Public
No Inheritance
Constructor with a reference to this object's associated command.

`GlStatus ExecuteFreeTextSearch ()`

Privilege: Public
No Inheritance
Used to search the COTS search engine for free text searches.

`GlStatus ExecuteKeywordSearch ()`

Privilege: Public
No Inheritance
Used to submit a query to the DBMS wrapper layer. The results are stored in the results list.

`GlStatus ExecuteSearch ()`

Privilege: Public
No Inheritance
Called to execute the search. Both a keyword and free-text search a executed as appropriate.

`GlStatus FormatResults (ostream *ostr)`

Privilege: Public
No Inheritance
Format the merged results into a HTML document to be returned to the WWW client across the HTTP connection. Write output to specified output stream.

`GlStatus MergeResults ()`

Privilege: Public
No Inheritance
Used to merge the results of a free text and keyword search. A list of URLS are writted to the HTML results list.

`void ~DsCtSearchCommand ()`

Privilege: Public
No Inheritance
Object destructor - Deep distruction, all associated results lists need to be freed. Call distructor for GlParameterList and lists of URLs.

5.5.2.7 Class DsDoClient

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
None

Description:

This object represnts the client for Document Data server.

Attributes:

`myClientName`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the client. Used to identify the protocol and behaviour of a WWW client.

`myClientVersion`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Version number of the client. Different WWW clients may have differnt behaviour and different versions may present a different interface.

`mySystemLog`

Privilege: Private
Data Type: GlLog &
Default Value: null
No Inheritance
Reference to the system log for exception reporting and logging.

Operations:

5.5.2.8 Class DsDoCommand

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
None

Description:

This object represnts the commands for Document Data.

Attributes:

`myCommandType`

Privilege: Private
Data Type: enum {}
Default Value: 0
No Inheritance
Type of command to execute. Relates to the service request type.

`myRequest`

Privilege: Private
Data Type: DsDoRequest &
Default Value: null
No Inheritance
Reference to the associated request.

Operations:

`void ExecuteCommand ()`

Privilege: Public
No Inheritance
Called to invoke the appropriate service object according to the command type.

5.5.2.9 Class DsDoRequest

Synopsis:

No Parent Class
Distributed Object
Is Associated With:
None

Description:

This object represents the requests for document data received vy the Docuement Data Server.

Attributes:

`myClient`

Privilege: Private
Data Type: DsDoClient &
Default Value: null
No Inheritance
Reference to the associated client for this request.

`myRequestName`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of this request - taken from the GlParameterList for this request.

`myRequestType`

Privilege: Private
Data Type: int
Default Value: null
No Inheritance
Type of request to be serviced.

`myServer`

Privilege: Private
Data Type: DsDoServer &
Default Value: null
No Inheritance
Reference to the associated server for this request.

Operations:

`void DsCIRequest ()`

Privilege: Public
No Inheritance
Object constructor - no specific implementation

`GlStatus ServiceRequest ()`

Privilege: Public
No Inheritance
Invoked for the execution of this service request, normally from the server after the client has constructed the request and called service request to the server.

`void ~DsCIRequest ()`

Privilege: Public
No Inheritance
Object destructor - call the destructor of the associated commands.

5.5.2.10 Class DsDoServer

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
None

Description:

This object represents server receiving requests for data from Data Server.

Attributes:

`myServerName`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the server object.

`myServerVersion`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Version of the server running.

`mySystemLog`

Privilege: Private
Data Type: GlLog &
Default Value: null
No Inheritance
Reference to the associated log file for fault and error logging.

Operations:

`GlStatus ShutDown ()`

Privilege: Public
No Inheritance
Shut down the server aborting outstanding requests.

`GlStatus StartUp ()`

Privilege: Public
No Inheritance
Start up server and return status indicaing readiness to service incomming requests.

5.5.2.11 Class DsEsAlgorithmDescriptionTypeID

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DsEsAlgorithmDescription (Aggregation)

Description:

This object represents the ID of the Algorithm Description type for the ESDT's.

Attributes:

`myTypeID`

Privilege: Private
Data Type: enum{DsEsAlgDescUn, DsEsSysDescDocTy, DsEsFilesDescDocTy, DsEsOpManTy, DsEsTestPiTy, DsEsATBDTy, DsEsDevStdDocTy, DsEsPrgGuideTy, DsEsPerTestresTy, DsEsDetDesDocTy};
Default Value: NOT IDENTIFIED
No Inheritance
This attribute is used to distinguish between the different Algorithm descriptions.

Operations:

`void DsEsAlgDescTyID (char *)`

Privilege: Public
No Inheritance
This operation provides the constructor for the ESDT's Algothrithm description.

`void GetTypeID ()`

Privilege: Public
No Inheritance
This operation retrieves the Type ID of the Algorithm Description.

`void SetTypeID ()`

Privilege: Public
No Inheritance
This operation sets the Type ID of the Algorithm description.

`void ~DsEsAlgDescTyID ()`

Privilege: Public
No Inheritance
This operation provides the desctructor for Type ID of the ESDT's Algorithm descriptions.

5.5.2.12 Class DsEsESDT

Synopsis:

Parent Class: DsGeESDTsof
Is Not A Distributed Object
Is Associated With:
Class: DsEsTypeID(Public)

Description:

This object represents the ESDT's of document type.

Attributes:

`$myDsEsESDTList`

Privilege: Private
Data Type: Date List<DsEsESDT *>
Default Value: NOT IDENTIFIED
No Inheritance

`myDocuementCreated`

Privilege: Private
Data Type: char *
Default Value: NOT IDENTIFIED
No Inheritance

`myDocumentUpdated`

Privilege: Private
Data Type: Date *
Default Value: NOT IDENTIFIED
No Inheritance

`myDocumentVersion`

Privilege: Private
Data Type: char *
Default Value: NOT IDENTIFIED
No Inheritance

`myDsEsCSDT`

Privilege: Private
Data Type: DsEsCSDT *
Default Value: NOT IDENTIFIED
No Inheritance

`myDsEsTypeID`

Privilege: Private
Data Type: DsEsTypeID *
Default Value: NOT IDENTIFIED
No Inheritance

`myFilePath`

Privilege: Private
Data Type: char *
Default Value: NOT IDENTIFIED
No Inheritance

`myTemplateName`

Privilege: Private
Data Type: char *
Default Value: NOT IDENTIFIED
No Inheritance

`myTemplateVersion`

Privilege: Private
Data Type: char *
Default Value: NOT IDENTIFIED
No Inheritance

`myURL`

Privilege: Private
Data Type: char *
Default Value: NOT IDENTIFIED
No Inheritance

Operations:

```
void DsEsESDT (DsEsTypeID *, DsEsCSDT *)
```

Privilege: Public
No Inheritance
Constructor for the ESDT document class.

```
void Externalize ()
```

Privilege: Protected Operation
No Inheritance
To Externalize the document data in a format to be used by the client.

```
void Internalize ()
```

Privilege: Protected Operation
No Inheritance
To internalize the document data for the ESDT's, in the server.

```
void Update ()
```

Privilege: Protected Operation
No Inheritance
Update the document data for the ESDT's.

```
void Virtual Validate ()
```

Privilege: Public
No Inheritance
Virtual validation of the class.

```
void ~DsEsESDT ()
```

Privilege: Public
No Inheritance
Destructor for the ESDT document data class.

5.5.2.13 Class DsEsGuideTypeID

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DsEsGuide (Aggregation)

Description:

This object represents the ID of the Guide type.

Attributes:

`myDsEsGuideTypeList`

Privilege: Private
Default Value: NOT IDENTIFIED
No Inheritance
This attribute lists the Guide type for the ESDT's.

`myTypeID`

Privilege: Private
Default Value: NOT IDENTIFIED
No Inheritance
Guide Type ID used to distinguish between different types of guide Document.

Operations:

`void DsEsGuideTypeID (char *Doctype)`

Privilege: Public
No Inheritance
This is the constructor for the Guide Type ID.

`void GetDsEsGuideType ()`

Privilege: Public
No Inheritance
This operation retrieves the Type of the Guides for the ESDT's.

`void SetDsEsGuideList (List<DsEsGuide *>)`

Privilege: Public
No Inheritance
This object sets the Guide Lists for the ESDT's.

`void ~DsEsGuidetypeID ()`

Privilege: Public
No Inheritance
This is the destructor for the Guide Type ID, for the ESDT's.

5.5.2.14 Class DsEsProductionPlanTypeID

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DsEsProductionPlan (Aggregation)

Description:

This object represents the ID of the type of the Production Plan.

Attributes:

`$myList`

Privilege: Private
Data Type: List<DsEsProductionPlanTypeID *>
Default Value: NOT IDENTIFIED
No Inheritance
List of currently active objects mainly used for fault recovery and memory management.

`myTypeID`

Privilege: Private
Data Type: enum{DsEsProductionPlanTyUn, DsEsProductionPlanBinaryTY, DsEsProductionPlanReportTy};
Default Value: NOT IDENTIFIED
No Inheritance
Production Plan used to distinguish between different types of Production Plans.

Operations:

`void GetTypeID ()`

Privilege: Public
No Inheritance
This operation retreives the Type ID for the Production Plans of the ESDT's.

`void SetTypeID ()`

Privilege: Public
No Inheritance
This operation sets the Type ID of the Production Plans for the ESDT's.

5.5.2.15 Class DsEsReferencePaperTypeID

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
DsEsReferencePaper (Aggregation)

Description:

This object represents the ID of the Reference Papers of ESDT's.

Attributes:

`$myDsEsRefPapTy`

Privilege: Private
Data Type: List<DsEsRefPapTyID *>
Default Value: NOT IDENTIFIED
No Inheritance
List of the currently active objects. Mainly used for fault recovery and memory management.

`myTypeID`

Privilege: Private
Data Type: enum{DsEsRefPapTyUn, DsEsElectJouTy, DsEsJouArt, DsEsStdADoc};
Default Value: NOT IDENTIFIED
No Inheritance
High level type ID used to distinguish between the main document types. Currently this includes Guides, Algorithm Descriptions, Reference Papers and Production plans.

Operations:

`void DsEsRefPapTyID (char * name)`

Privilege: Public
No Inheritance
Cosnstructor for the Reference paper Type ID.

`void GetTypeID ()`

Privilege: Public
No Inheritance
This operation retrieves the Type ID for the Reference Paper.

`void SetTypeID ()`

Privilege: Public
No Inheritance
This operation sets the Reference Paper Type ID.

5.5.2.16 Class DsEsTypeID

Synopsis:

No Parent Class
Is Not A Distributed Object
Is Associated With:
Class: DsEsESDT(Public)

Description:

This object represents the ID of the type of the ESDT.

Attributes:

`myDsEsTypeList`

Privilege: Private
Data Type: List<DsEsTypeID *>
Default Value: NOT IDENTIFIED
No Inheritance
Lists the ESDT types for the Document Type ESDT's.

`myTypeID`

Privilege: Private
Data Type: enum{DsEsTyUn, DsEsGuTy, DsEsrefTy, DsEsAlgDesTy, DsEsPrPITy};
Default Value: NOT IDENTIFIED
No Inheritance
This attribute represents the Type ID for ESDT's.

Operations:

`void DsEsTypeID (char *)`

Privilege: Public
No Inheritance
This is the constructor for the type ID's.

`void GetEsDsType ()`

Privilege: Public
No Inheritance
Retrieves the type ID for the Document type ESDT's.

`void SetDsEsType ()`

Privilege: Public
No Inheritance
Sets the Type ID for the document type ESDT's.

`void ~DsEsTypeID ()`

Privilege: Public
No Inheritance
This is the destructor for the ESDT's types.

5.5.2.17 Class DsSvServer

Synopsis:

Parent Class: DsDoServer
Is Not A Distributed Object
Is Associated With:
Class: CallingObject(Private)
Class: DsCtRequest(Public)
Class: DsSeIndexer(Private)
Class: DsCtClient(Public) manages
Class: DsCtClient(Public) serves
Class: DsCtRequest(Public) services

Description:

The Object represents the server object for Document Data Server.

Attributes:

`$myServerList`

Privilege: Private
Data Type: List <DsSvServer>
Default Value: null
No Inheritance
List of currently active servers. Used for fault recovery and memory management.

`myClientList`

Privilege: Private
Data Type: List <DsCtClient>
Default Value: null
No Inheritance
List of the active clients being serviced by the server.

`myHostAddress`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance

`myIPNumber`

Privilege: Private
Data Type: int
Default Value: 0
No Inheritance
IP number of the server.

`myPortNumber`

Privilege: Private
Data Type: int
Default Value: 0
No Inheritance
TCP/IP port number on the host to monitor for incoming HTTP requests.

`myProtocolName`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the communication protocol supported by the server.

`myProtocolVersion`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Version of the communication protocol supported by the server.

`mySecurityProtocol`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Name of the security protocol used by the server. Needs to be compatable with the client security protocol for a secure connection.

`mySecurityProtocolVersion`

Privilege: Private
Data Type: char *
Default Value: null
No Inheritance
Version of the security protocol used by the server.

`myServerName`

Privilege: Private
Data Type: char *
Default Value: null
Inherited From: DsDoServer
Name of the server object.

`myServerVersion`

Privilege: Private
Data Type: char *
Default Value: null
Inherited From: DsDoServer
Version of the server running.

`mySystemLog`

Privilege: Private
Data Type: GlLog &
Default Value: null
Inherited From: DsDoServer
Reference to the associated log file for fault and error logging.

Operations:

`void DsDoServer ()`

Privilege: Public
No Inheritance
Object constructor - no specific implementation

`GlStatus Listen ()`

Privilege: Private
No Inheritance
Monitors the TCP/IP port for incomming requests.

`GlStatus ServiceRequest ()`

Privilege: Public
No Inheritance
Called by client to execute a service request.

`GlStatus ShutDown ()`

Privilege: Public
No Inheritance
Shut down of an active server. Currently active clients will have their associated service requests aborted.

`GlStatus StartUp ()`

Privilege: Public
No Inheritance
Start up server. Return status to calling object indicating the server is ready to service incoming requests.

`void ~DsDoServer ()`

Privilege: Public
No Inheritance
Object destructor - destroy associated client objects

`GlStatus ShutDown ()`

Privilege: Public
Inherited From: DsDoServer
Shut down the server aborting outstanding requests.

`GlStatus StartUp ()`

Privilege: Public
Inherited From: DsDoServer
Start up server and return status indicaing readiness to service incomming requests.